
MINNESOTA
EDUCATIONAL
COMPUTING
CONSORTIUM

# INTERMEDIATE APPLESOFT™ BASIC

#660

# INTERMEDIATE

# APPLESOFT ™

# BASIC

Reasonable efforts have been made in the preparation of this courseware to ensure its accuracy. MECC cannot assume any liability resulting from errors, omissions, or use of the information contained herein.

This booklet is a product of:

Minnesota School Districts
Data Processing Joint Board
St. Paul, Minnesota



January, 1981

Latest Printing: August 1, 1983

# MECC TRAINING MATERIALS FOR THE APPLE II[TM] MICROCOMPUTER

## Preface

This booklet is one of a series providing training materials on various aspects of using the APPLE II microcomputer in education. This material is designed primarily for use in the "teaching of teachers". It would most naturally be used in workshop or class sessions for educators desiring to gain expertise on the APPLE II so they could in turn use this microcomputer in their elementary, secondary, or college classes. However, the multi-purpose design of these materials also makes them appropriate for self-instructional use.

The assumption is made that the trainer using these materials already has an adequate knowledge of the subject and does not have to rely on the materials themselves to learn about the APPLE II. Rather, the materials save the trainer the time needed to organize a workshop or course and to create handouts and exercises for the participants.

Each booklet was prepared by someone who used the materials in teaching a course and thus it will reflect that person's own philosophy on how topics should be organized and presented. Trainers will have individual styles and may want to alter the materials to some degree to fit their own preferences in teaching.

While these booklets are published by the Minnesota Educational Computing Consortium (MECC), they are the result of a joint effort contributed to by various participants in the Consortium, including MECC staff, the TIES organization, and the University of Minnesota Computer Center. For more information on all MECC support activities, including those for the APPLE II microcomputer, see the Appendix on MECC Instructional Services.

# TABLE OF CONTENTS
# AND COURSE OUTLINE

# MECC TRAINING MATERIALS FOR THE APPLE II MICROCOMPUTER

## Intermediate Applesoft BASIC

### Introduction

These materials are intended to be a continuation of MECC Training Materials for Beginning Applesoft BASIC. Beginning topics are reviewed and extended. New topics, prerequisite to an advanced BASIC course, are introduced. The materials were designed to be self-instructional materials as well as a guide for teacher workshops or student classes.

The package includes four instructional units and teacher notes.

Each unit contains sample programs to serve as introductory material. Each unit also contains an activity section to reinforce and extend topics introduced. It is assumed that at least one Apple II microcomputer will be available to demonstrate sample programs.

The TEACHER NOTES section contains comments and possible solutions to sample programs and listings and sample runs of suggested solutions to the activities.

The trainer may wish to prepare a demonstration diskette for use with these materials. Directions are given on Page T-1 of the Teacher Notes section.

Before using these materials the participant should:

1. Complete the Beginning Applesoft BASIC Training Materials or a similar beginning BASIC course.
2. Gain background experience with the Apple microcomputer by using programs on disk.

After completing these materials the participant should be able to:

1. Use a flow chart to aid in the development process.
2. Use Applesoft system commands to manipulate and modify programs saved on disk.
3. Design programs that are easy to read, understand, modify and run.
4. Format output to the screen in a readable and user-controlled manner.

### Equipment

It is generally assumed in the use of these materials that the participants are using the Apple II with Autostart, Applesoft in ROM and a disk drive. Each participant should have a blank diskette and access to a DOS 3.3 System Master Diskette.

### Acknowledgements

These materials were developed by Marcia Horn and the TIES Instructional Staff.

## Operating the Apple

If you are not familiar with the Apple (or you feel you need to be reacquainted), perform the following procedures before continuing with this packet.

### TURNING ON THE APPLE - "Booting the System"

Step 1: Turn on the monitor or T.V. (UHF Channel 33).

Step 2: Put a disk in the disk drive (label side up and toward you) and close the drive door.

Step 3: Ground yourself by touching the metal on the back of the Apple.

Step 4: Turn on the Apple (switch is on the back, to the left of the power cord).

### USING PROGRAMS FROM A DISK WITHOUT A MENU

Step 1: Boot the system (see steps 1-4 above).

Step 2: If a list of program names is not on the screen and you have a ] prompt, type:
CATALOG
and press RETURN key.

Step 3: Type: LOAD program name
(ex. LOAD OREGON)

Type: RUN

OR

Type: RUN program name
(ex. RUN OREGON).

OR

### ...FROM A DISK WITH A MENU

Step 1: Boot the system (steps 1-4 above).

Step 2: If a list of program names and the message, "TYPE THE NUMBER OF THE PROGRAM YOU WANT", is on the screen then you have a disk with a menu.

Step 3: Type the number of the program and press RETURN.

### CHANGING TO ANOTHER PROGRAM ON THE SAME DISK

Step 1: If you have the ] prompt or the disk menu follow steps 2 and 3 for using programs from a disk.

Step 2: If you are in the middle of a program and wish to change to another program, press RESET, type PR#6 and press RETURN. Follow steps 2 and 3 for using programs from a disk.

### USING PROGRAMS ON A DIFFERENT DISK

Step 1: Make sure the disk drive "in use" light is off.

Step 2: Put in the desired disk, returning the other disk to its paper envelope.

Step 3: Follow steps 1 and 2 for changing to another program on the same disk.

### TURNING OFF THE APPLE

Step 1: Make sure the disk drive light is off.

Step 2: Remove the disk, put it in its paper envlope and store it vertically.

Step 3: Turn off the T.V. and the Apple

## Initializing a Disk

To prepare a new disk for storing programs, you must first initialize the disk. The initializing process writes DOS (Disk Operating System) onto the disk, sets up the catalog area and marks the 560 sectors where programs and files will be stored. Initializing a disk containing programs will "wipe it clean" of all old programs.

Step 1: With the Apple power turned off, place the DOS 3.3 System Master in the disk drive.

Step 2: Turn on the Apple and wait for the following to appear on the screen.

```
DOS VERSION 3.3                 08/25/80
APPLE II PLUS OR ROMCARD        SYSTEM MASTER

]
```

Step 3: Take out the System Master and put in the disk to be initialized.

Step 4: Type the following, pressing RETURN after each line:

```
]NEW
]10 HOME
]20 PRINT"THIS DISK BELONGS TO  (your name)  "
]30 END
]INIT HELLO
```

Wait about 30 seconds until the disk drive light goes off.

Step 5: Check your disk by turning off the Apple, placing your new disk in the drive and then turning on the Apple. The screen should clear and the following should appear:

```
THIS DISK BELONGS TO  (your name)
]
```

Step 6: Remove the disk, put it in its paper envelope and write your name on the label using a felt tip pen.

---

*** NOTE: A disk initialized on a 32K Apple will work on either a 32K or 48K system. A disk initialized on a 48K system will only work on a 48K system. Use the program MASTER CREATE on the System Master disk to convert the disk into a disk that will work on a 32K Apple. (MASTER CREATE does not alter any files that may be saved on the disk. It just changes DOS.)

## Disk Format

The initializing process writes DOS (Disk Operating System) onto the disk. DOS contains the information needed by the Apple to operate a disk drive.

The directory or catalog of the disk contents is also placed on the disk during the initialization. The directory contains information concerning the names and locations of files on the disk.

In addition, the initializing process divides the disk into tracks and sectors where files will be stored by the user.

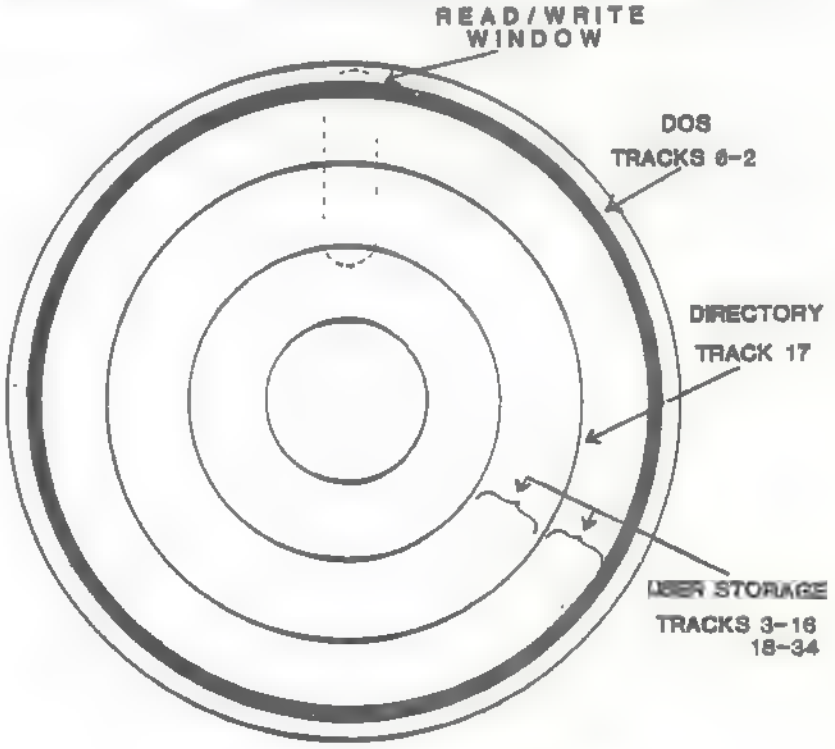
### Tracks

An initialized disk has 35 concentric tracks. The tracks are numbered 0 to 34, beginning with the outermost track.

DOS is stored on tracks 0, 1 and 2.

The directory is stored on track 17.

Of the 35 tracks, 31 tracks remain for the user to store files. The user tracks are 3-16 and 18-34.

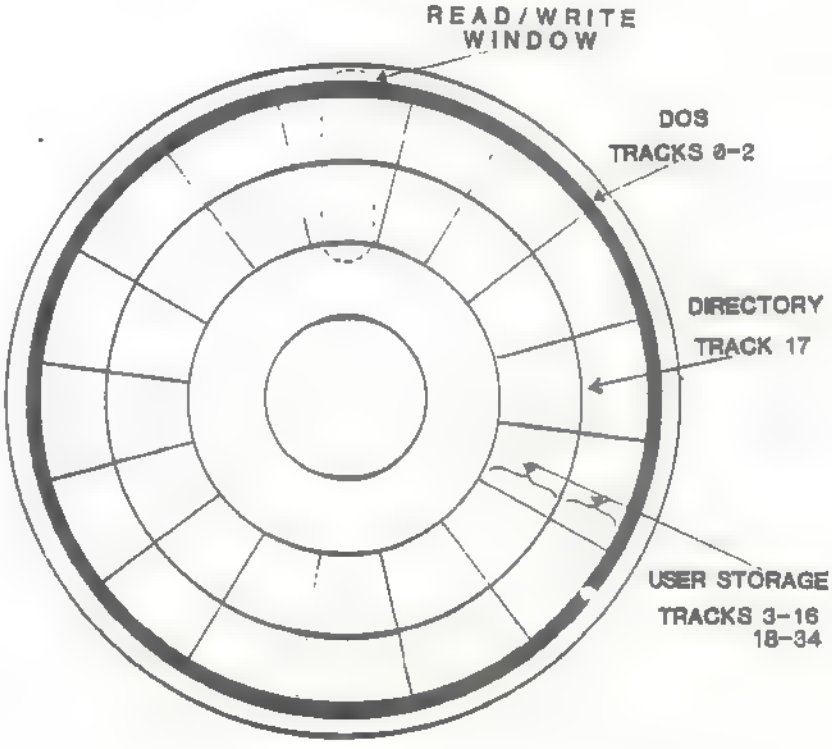



### Sectors

Each track is further divided into 16 sectors. The phrase "16 sector disk" means each track has 16 sectors.

There are 560 sectors on a disk. DOS and the directory take up 64 sectors, leaving 496 sectors for user storage.

Each sector can store up to 256 bytes or characters of information. The 560 sectors have the capacity of storing 143K or 143,360 bytes of information.



## System Commands

System commands are direct instructions to the computer. They are used to manipulate programs for utility purposes. System commands execute immediately and are generally not used with line numbers as parts of programs.

### CATALOG

Put the DOS 3.3 System Master in the disk drive. Turn on the Apple. When you have the ] prompt type:

CATALOG

The Apple will display a catalog listing of programs stored on the disk.

```
]CATALOG

DISK VOLUME 254

*A 006 HELLO
*I 018 ANIMALS
*T 003 APPLE PROMS
*I 006 APPLESOFT
*I 026 APPLEVISION
*I 017 BIORHYTHM
*B 010 BOOT13
*A 006 BRIAN'S THEME
*B 003 CHAIN
*I 009 COLOR DEMO
*A 009 COLOR DEMOSOFT
*I 009 COPY
*B 003 COPY.OBJ0
*A 009 COPYA
*A 010 EXEC DEMO
*B 020 FID
*B 050 FPBASIC
*B 050 INTBASIC
```

Press any key (except RESET) to send up the rest of the catalog.

```
*A 028 LITTLE BRICK OUT
*A 003 MAKE TEXT
*B 009 MASTER CREATE
*B 027 MUFFIN
*A 051 PHONE LIST
*A 010 RANDOM
*A 013 RENUMBER
*A 039 RENUMBER INSTRUCTIONS
*A 003 RETRIEVE TEXT

]
```

The Applesoft prompt "]" lets you know that all of the catalog has appeared.

# * A 009 COLOR DEMOSOFT

"*" indicates the file is locked.

"A" indicates the file type is Applesoft. Others are:
"I" - Integer
"B" - Binary
"T" - Text

"009" indicates the program occupies 9 sectors on the disk. The total number of sectors available for user storage if 496.

"COLOR DEMOSOFT" is the program name.

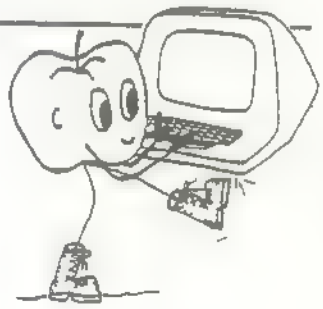
## Booting the System

Two ways of "booting the system" or "booting DOS" (Disk Operating System) are:

1. Turn the Apple off. Put in a disk. Turn the Apple on.
2. Press RESET. Put in a disk. Type PR#6. Press RETURN.

When the system is booted, the Apple is first "taught" how to operate with a disk drive (DOS is loaded into the computer's memory). After DOS is loaded, the Apple automatically runs the program typed in when the disk was initialized (the program named HELLO on your disk).

---



Put your disk in the disk drive and type PR#6. Press RETURN.

Type the following: CATALOG

Answer these questions:

- Is the HELLO program locked? _____
- How many sectors does the HELLO program occupy? _____
- How many sectors are remaining for storing other programs and files on your disk? _____
- What type of file is the HELLO program? _____

| Type the following. Press RETURN after each line. | Record what happens below. |
|---|---|
| LIST | |
| RUN | |
| NEW | |
| LIST | |
| RUN | |

Typing NEW clears any previously used program from the Apple's memory. After the system is booted, the HELLO program is in the Apple's memory. Loading and running another program stores that program in the Apple's memory. Always type NEW before typing in a new program or you'll end up with two programs mixed together.

## Saving Programs on Disk

| Type the following: | This is what happens: |
|---|---|
| NEW | The Apple's memory is cleared. |
| 10 HOME<br>20 PRINT "2 + 8 ="<br>30 PRINT 2 + 8<br>40 END | One line of this program is placed into memory each time RETURN is pressed. |
| HOME | The screen is cleared. |
| LIST | The program in memory is listed on the screen. |
| RUN | The program in memory is executed. |
| SAVE PROGRAM EXAMPLE | The program in memory is saved on the disk by the name following the word SAVE. |
| CATALOG | PROGRAM EXAMPLE is now on the disk catalog. |

The name given to a program when saving it on disk must begin with a letter and can be up to 30 characters long.

Commas and semicolons cannot be used in a program name.
Control characters should not be used in a program name.

| Type the following: | This is what happens: |
|---|---|
| LIST | The program in memory is listed on the screen. |
| NEW | Memory is cleared. |
| LIST | Nothing is listed since memory was just cleared when NEW was typed. |

## Renaming and Altering Programs Saved on Disk

(UNDERLINED words are the system commands.)

| Type the following: | This is what happens: |
|---|---|
| LOAD PROGRAM EXAMPLE | The Apple loads a copy of the program, PROGRAM EXAMPLE, from the disk into the Apple's memory. |
| RUN | The Apple executes the program in memory. |
| RUN PROGRAM EXAMPLE | The Apple loads a copy of PROGRAM EXAMPLE from the disk and executes it. |
| RENAME PROGRAM EXAMPLE, EXAMPLE<br>CATALOG | The Apple changes the program's name on the disk catalog from PROGRAM EXAMPLE to EXAMPLE. |

You may wish to alter the version of the program that you saved on disk. As you work on long programs or as you find bugs in programs, lines on programs already saved on disk may need to be altered. The following example illustrates the process.

| Type the following: | This is what happens: |
|---|---|
| LOAD EXAMPLE<br>LIST | A copy of the program EXAMPLE is loaded into memory and then listed on the screen. |
| 40 PRINT "TYPE A NUMBER"<br>50 INPUT A<br>60 PRINT "TYPE ANOTHER"<br>70 INPUT B<br>80 PRINT "THE SUM IS "A + B<br>90 END | These lines are put into memory with EXAMPLE program. |
| LIST | The entire program in memory is listed. |
| RUN | The program is executed. Type numbers when asked to do so. |
| SAVE EXAMPLE | The current program in memory replaces the old version of EXAMPLE on the disk. |

## System Commands

### Locking, Unlocking and Deleting Programs

| Type the following: | This is what happens: |
|---|---|
| LOCK EXAMPLE<br>CATALOG | An asterisk (*) will appear to the left of the program name. |
| DELETE EXAMPLE | The Apple "beeps" and prints FILE LOCKED. The Apple will attempt to delete the program EXAMPLE from the disk, but will find out that it is locked. EXAMPLE is not deleted. |
| NEW<br>1Ø HOME<br>2Ø PRINT "APPLES ARE GREAT! "<br>3Ø END<br>SAVE EXAMPLE | The Apple "beeps" and prints FILE LOCKED. The Apple will attempt to replace the old program EXAMPLE with this new version but will find out that it is locked. EXAMPLE is not replaced. |
| SAVE GREAT<br>CATALOG | The new program is saved by the name GREAT. |

Locking a program is a way to protect the program from being accidentally deleted or lost when saving another program by the same name.

When you initialized your disk on page I-2, the program created and used in the initialization process was named HELLO (see Step 4: INIT HELLO). When the system is booted with your disk, the Apple will look for a program with the name HELLO.

It is a good idea to lock the program used to initialize a disk.

Type: LOCK HELLO
Type: CATALOG

You should see an asterisk to the left of HELLO.

| Type the following: | This is what happens: |
|---|---|
| UNLOCK EXAMPLE<br>CATALOG | The asterisk (*) disappears. |
| DELETE EXAMPLE<br>CATALOG | The program EXAMPLE is deleted from the disk. |

Locking is not permanent. A file needs to be unlocked before changes may be made in the

# INVERSE, FLASH, NORMAL and SPEED

## INVERSE, FLASH and NORMAL

The INVERSE or FLASH command may be used to draw attention to directions, provide reinforcement or give warnings during the run of a program.

INVERSE causes all printing that follows to be output to the screen as black characters on a white background.

FLASH causes all printing that follows to be output to the screen alternating from white characters on black to black characters on white.

NORMAL sets the screen output back to the usual white characters on a black background. A NORMAL statement should follow the PRINT statements that are flashed or output in inverse on the screen. *** If NORMAL is not included, all prints, listings and runs will continue in the last video mode. LOAD SAMPLE I-9 from the Intermediate BASIC disk or type the following:

```
NEW

 5  HOME
 10  PRINT "WHAT IS YOUR FIRST NAME ";
 20  INPUT A$
 30  PRINT "HI THERE "A$
 40  INVERSE
 50  PRINT "HAPPY DAY TO YOU"


 60  NORMAL
 70  PRINT "WHAT IS YOUR LAST NAME, "A$;

 80  INPUT B$
 85  HOME

 90  FLASH

100  PRINT "HELLO "A$,B$

105  NORMAL

110  END
```

Line 40 sets the video mode so that all PRINT statements that follow are printed in black on a white background.

Line 60 returns the video mode to white letters on black background.

Line 85 clears the screen.

Line 90 sets the video mode so that all PRINT statements that follow flash on the screen.

Line 105 returns the video mode to white letters on black background.

Run the program.

## SPEED

Add this line to the program: 7 SPEED = 50

Run the program.

The speed of printing can be set from 0 through 255. SPEED = 0 is the slowest setting. SPEED = 255 is the normal printing speed. All runs and listings of programs will continue at the last speed set. Type: LIST

Want things back to normal? Type: SPEED = 255
LIST

***Whenever the speed is altered in a program, include a line to set printing back to normal before the program's end. Add the following line:

107 SPEED = 255

Run the program.
List the program.

## Packing Statements

LOAD SAMPLE I-10 from the Intermediate BASIC disk or type the following:

```
NEW

10  HOME
20  INPUT "WHAT IS YOUR NAME?";A$
30  PRINT
40  PRINT "HELLO "A$
50  END
```

Run the program.

---

The colon is used to pack statements together using one line number.

Change line 20 as follows:

```
20 INPUT "WHAT IS YOUR NAME?"; A$  :  PRINT  :  PRINT "HELLO  "A$
```

Take out lines 30 and 40.

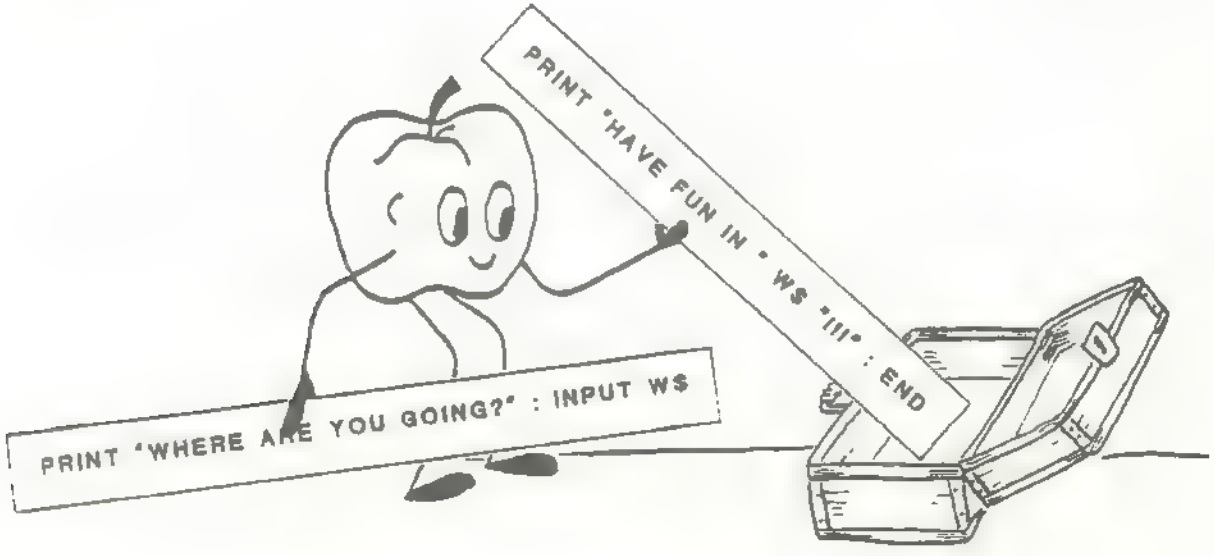
Run the program.
List the program. Compare the listing to the listing above.

---

To save computer storage space for a program and to speed up execution of a program, several lines may be condensed to one line of program. Statements that work together to perform a task are commonly grouped as one line of program.

Since there is only one line number the program can only branch to the first statement in the line. A GOTO cannot send the program to the middle of a packed statement. Correcting and modifying these long packed statements means retyping the entire line. Reading and following the logic of programs with many packed statements can be difficult.

A single statement or packed statement may be at most 239 characters long, including the line number.

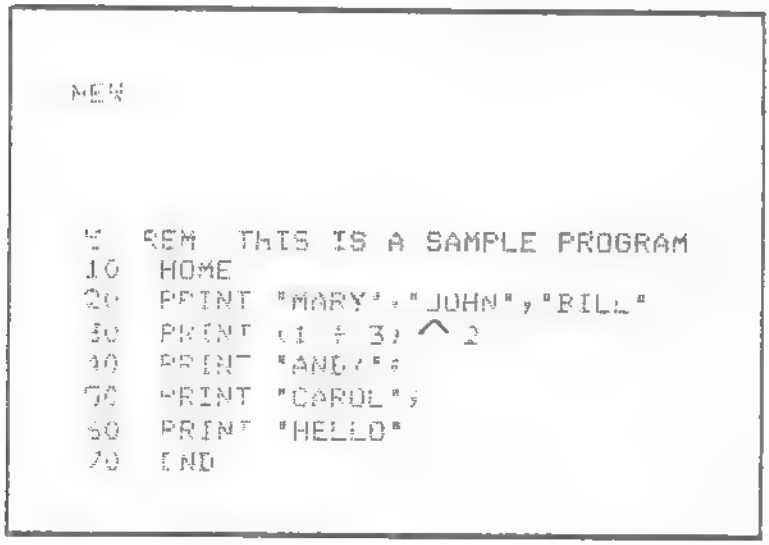


## Activities

***NOTE: Activity programs I-7 are review of topics included in Beginning Applesoft BASIC training materials.

### Program 1: REM, PRINT, GOTO

1. LOAD PROGRAM I-1 from the Intermediate BASIC disk or type in the following:




2. Run the program.

3. Change lines 40 and 50 to include commas instead of semicolons:

```
40 PRINT "ANDY",
50 PRINT "CAROL",
RUN
```

How does the comma change the output? _____

4. Add this line:

```
65 GOTO 60
RUN
```

(To stop, hold down the CTRL key while typing C.)

## Activities

### Program 2: LET

1. LOAD PROGRAM I-2 from the Intermediate BASIC disk or type in the following:

```
NEW

 10  HOME
 20  LET A = 4
 30  LET B = 5
 40  LET C = A + B
 50  PRINT C
 60  LET PRODUCT = A * B
 70  PRINT PRODUCT
 80  END
```

2. Run the program.

3. Change the following lines, leaving out the word LET:

```
2Ø  A = 4
3Ø  B = 5
6Ø  PRODUCT = A*B
RUN
```

Do you notice any changes in the output? _____

(The word LET is optional in Applesoft BASIC.)

4. Change the variable name in the following line:

```
7Ø  PRINT PRICE
RUN
```

Do you notice any changes in the output? _____

(A variable name may be up to 238 characters long, but the Apple only uses the first two characters to distinguish them. In the program above, PRODUCT and PRICE are the same variable name.)

## Activities

### Program 3: Checking for 2 Character Variable Names

1. LOAD PROGRAM I-3 from the Intermediate BASIC disk or type in the following:

```
NEW

 5  HOME
10  C = 3
15  CA = 15
20  CAB = 5
30  CAD = 6
31  CAC = 7
32   PRINT C
33   PRINT CA
34   PRINT CAB
35   PRINT CAD
37   PRINT CAC
50   END
```

2. Run the program. This is what is happening in memory:

   Line 10 assigns the value 3 to C.

   Line 15 assigns the value 15 to CA.

   Line 20 assigns the value 5 to CA (the Apple only uses the first two characters to distinguish variable names. Line 20 replaces the value assigned to CA by line 15).

   Line 30 assigns the value 6 to CA.

   Line 31 assigns the value 7 to CA.

   Line 32 prints the value of C.

   Line 33 prints the value of CA.

   Line 34 prints the value of CA (the Apple only uses the first two characters in the variable name).

   Line 35 prints the value of CA.

   Line 37 prints the value of CA.

3. Delete line 20 by typing 20 and pressing RETURN.
   Add this line:

   ```
   36 CAB = 4
   RUN
   ```

   How is the output different? _____

   Why? _____

## Activities

### Program 4: INPUT

1. LOAD PROGRAM I-4 from the Intermediate BASIC disk or type in the following:

```
NEW


10  HOME
20  PRINT "WHAT IS YOUR NAME ";
30  INPUT B$
40  PRINT "HI, " B$ "."
50  LET C$ = "THIS IS YOUR LUCKY DAY, "
60  PRINT C$;B$ "."
70  END
```

**Notice that string variables used (B$ and C$) do not first have to be dimensioned as in other forms of BASIC. Up to 256 characters may be saved as a string variable (line 50).

2. Run the program.

3. Change the following line:

   20 INPUT "WHAT IS YOUR NAME"; B$

   Take out line 30 by typing 30 and pressing RETURN.

   Run the program.

   Do you notice anything different in the output? _____

   Do you have a question mark after the question from line 20? _____

   Change line 20 to make a question mark appear.

** INPUT statements without messages automatically send a question mark to the screen: 30 INPUT B$

** When a message is included with an INPUT, no question mark is sent to the screen: 30 INPUT "WHO ARE YOU"; B$

## Activities

### Program 5: IF-THEN

1. LOAD PROGRAM I-5 from the Intermediate BASIC disk or type in the following:

```
NEW

 10  HOME
 20  LET T = 0
 30  PRINT "TYPE 0 TO HAVE THE TOTAL PRINTED."
 40  INPUT "TYPE A NUMBER:  " ; N
 50  IF N = 0 THEN 80
 60  LET T = T + N
 70  GOTO 30
 80  PRINT "TOTAL = "T
 90  END
```

2. Run the program.

3. Change the following line, using GOTO instead of THEN:

```
50  IF N = 0  GOTO 80
RUN
```

   Are there any differences in the run of the program? _____

4. Change the program to have -1 be the number to type to have the total printed.

5. Add lines to have the Apple count the numbers that are typed in and then print the average.

## Activities

### Program 6: READ, DATA

1. LOAD PROGRAM I-6 from the Intermediate BASIC disk or type in the following:

```
NEW

 10  HOME
 20  READ A,B
 30  PRINT A,B
 40  READ C$,D$
 50  PRINT C$,D$
 60  DATA  5,6,ANN,ANDY
 70  END
```

2. Run the program.

3. Change this line to include quotation marks:

```
60  DATA 5, 6, "ANN", "ANDY"
RUN
```

Do you notice anything different about the output? _____

(You do not need to include strings within quotation marks in DATA statements as you do in other BASICS. However, if you wish to use a comma within a string, the string must be enclosed in quotation marks.)

4. Change this line:

```
60  DATA 5, 6, ST. PAUL, MN, HUNTLEY, MN
RUN
```

What happened to HUNTLEY, MN? _____

Change the line to include quotation marks:

```
60  DATA 5, 6, "ST. PAUL, MN", "HUNTLEY, MN"
RUN
```

Did you see HUNTLEY, MN this time? _____

## Activities

### Program 7: FOR-NEXT Loops

1. LOAD PROGRAM I-7 from the Intermediate BASIC disk or type in the following:

```
NEW


10  HOME
20  FOR NUMBER = 0 TO 12
30  PRINT "NUMBER = "NUMBER
40  NEXT NUMBER
50  END
```

2. Run the program.

3. Change the following line to include STEP:

```
20  FOR NUMBER = Ø TO 12 STEP 2
RUN
```

What is different about the output? _____

Change the following again:

```
20  FOR NUMBER = 12 TO Ø STEP -1
RUN
```

What is different about the output? _____

## Activities

### Program 8: INVERSE, FLASH, NORMAL and SPEED

1. LOAD PROGRAM I-8 from Intermediate BASIC disk or type the following:

```
NEW


 2  HOME
 5  SPEED= 0
 10  PRINT "THIS IS AS SLOW AS I CAN GO."
 20  SPEED= 150
 30  PRINT "SPEED 150 IS NICE FOR READING."
 40  SPEED= 255
 50  PRINT "THIS IS NORMAL SPEED."
 60  END
```

2. Run the program.

3. The speed of printing can be set from 0 through 255. Experiment with other speeds by changing lines 5, 20 and 40.

4. Add lines to make the message, "THIS IS AS SLOW AS I CAN GO", print in inverse and the message, "THIS IS NORMAL SPEED", flash on the the screen. Record the lines added in the space below.

5. Run the program.

## Loops

FOR and NEXT statements provide a simple and efficient method for repeating a sequence of instructions. The FOR statement labels the beginning of the loop. The NEXT statement labels the end of the loop. All statements between the FOR and NEXT statements are called the body of the loop. The body of the loop is the sequence of instructions that is to be repeated. Both programs below will print the square of each number 1 through 12. Notice that the FOR NEXT version is shorter.

(Using LET and IF THEN)

```
10  HOME
20  PRINT "NUMBER","SQUARE"
30  PRINT "------","------"
40  PRINT
50  LET N = 1
60  PRINT N,N * N
70  LET N = N + 1
80  IF N > 12 THEN 100
90  GOTO 60
100  PRINT "I AM FINISHED LOOPING!"
110  END
```

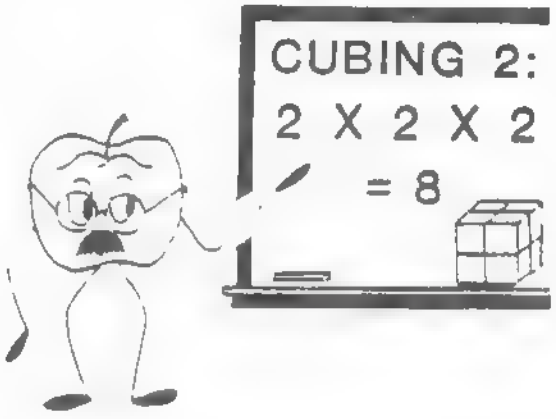
(Using FOR NEXT Loop)

```
10  HOME
20  PRINT "NUMBER","SQUARE"
30  PRINT "------","------"
40  PRINT
50  FOR N = 1 TO 12
60  PRINT N,N * N
90  NEXT N
100  PRINT "I AM FINISHED LOOPING!"
110  END
```

SAMPLE II-1A from the Intermediate BASIC disk.

1. Type in the FOR NEXT version of the squaring program or LOAD SAMPLE II-1B from the Intermediate BASIC disk and run it.

2. Change 3 lines to make the program print all the cubes (N*N*N) of the numbers 11 through 20. Record the lines you changed in the space below:




3. Change this line to print cubes of even numbers 2 through 20:

```
50 FOR N = 2 TO 20 STEP 2
RUN
```

4. Change this line to print all the cubes starting with 10 and ending with 0:

```
50 FOR N = 10 TO 0 STEP -1
RUN
```

## Loops

In the squaring and cubing program the loop variable (N) was used in the body of the loop. The value of the loop variable should not be altered in the body of the loop by a LET statement (LET N = N-1)...or the results will be quite strange. Try this program or LOAD SAMPLE II-2A from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  FOR N = 1 TO 5
30  PRINT N
40  LET N = N - 1
50  NEXT N
60  END
```

Run the program.

(To stop, hold the CTRL key down while typing C.)

---

The loop variable does not have to be used at all in the body of the loop. It may be used only to control the number of times the loop is repeated. Type in this program or LOAD SAMPLE II-2B from the Intermediate BASIC disk:

```
]NEW


10  HOME
20  FOR A = 1 TO 5
30  PRINT "PROGRAMMING IN BASIC"
80  NEXT A
90  END
```

Run the program.

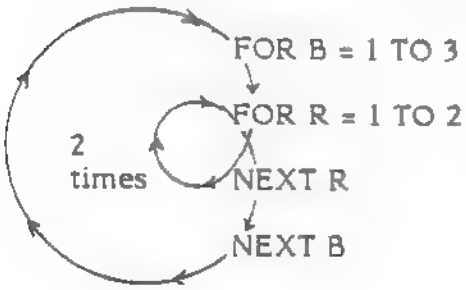
Add these lines:

```
40  FOR B = 1 TO 2
50  PRINT "IS FUN"
60  NEXT B
RUN
```

This is an example of a nested loop. Loop B is inside loop A. Each time that loop A is completed, loop B is repeated 2 times.

## Nested Loops

Loops may be placed within loops...but loops must never overlap. A way to check the nesting of loops is to draw lines connecting the FOR statements with their NEXT statements. These lines should never cross each other. Type the following or LOAD SAMPLE II-3 from the Intermediate BASIC disk:




```
]NEW


 10  HOME
 20  PRINT "THIS WILL PRINT 3 BLOCKS"
 30  PRINT "WITH 2 ROWS OF 4 STARS."
 40  PRINT
 50  FOR B = 1 TO 3
 60  PRINT "BLOCK #"B
 70  FOR R = 1 TO 2
 80  PRINT "****"
 90  NEXT R
100  PRINT
110  NEXT B
120  END
```

Run the program.

Make these changes:

```
90  NEXT B
110  NEXT R
RUN
```

What happens? _____

Why? _____

Change the lines back to:

```
90  NEXT R
110  NEXT B
RUN
```

Add two lines to make the APPLE repeat drawing the 3 blocks two times. Record the lines you add in the space below:

(HINT: Also add line 11 SPEED = 175 and line 119 SPEED = 255.)

# READ-DATA

READ-DATA statements are useful when you wish to store data for variables within a program. This is easier than typing the data in for INPUT statments each time the program is run. It may also be more efficient than setting the value for each variable with a LET statement.

The READ statement contains the variable in which the string or number in the DATA statement will be stored. Items in DATA statements must occur in the order in which the READ statements will ask for them. The Apple uses a "pointer" which moves from one DATA item to the next as the READ statements call for them. Type in the following program or LOAD SAMPLE II-4 from the Intermediate BASIC disk:

```
]NEW

 10  HOME
 20  PRINT "SCORES ON 20 PT. QUIZ"

 30  PRINT "POINTS","PERCENT"
 40  PRINT "------","-------"
 50  FOR S = 1 TO 4
 60  READ P
 70  PRINT P,P / 20 * 100
 80  NEXT S
 90  END
 100  DATA  20,0,15,17


]RUN

SCORES ON 20 PT. QUIZ
POINTS          PERCENT
------          -------
20              100
0               0
15              75
17              85

]
```

Run the program.

Each time the Apple travels through the S loop and reads a value for P the pointer in the DATA statement moves to the next item as illustrated with the ⬇ below:

Loop with S = 1:

```
100 DATA  20, 0, 15, 17
          ⬇
```

Loop with S = 2:

```
100 DATA  20, 0, 15, 17
              ⬇
```

Loop with S = 3:

```
100 DATA  20, 0, 15, 17
                  ⬇
```

Loop with S = 4:

```
100 DATA  20, 0, 15, 17
                      ⬇
```

Change the following line to make the program loop 5 times:

```
50 FOR S = 1 TO 5
RUN
```

What message does the Apple give you? _____

(The DATA pointer ran out of numbers when trying to READ a value for P in line 60.)

## READ and DATA

If more than one variable is used in a READ statement the DATA pointer moves as many positions as is needed to get values for each variable. Care must be taken to make sure the items in the DATA statements match the variable type in the READ statements (e.g., N$ for a string, A for a number). Type the following or LOAD SAMPLE II-5 from the Intermediate BASIC disk:

```
]NEW

 10  HOME
 20  PRINT "PROGRAM TO AVERAGE"
 30  PRINT "PTS. ON 3 TESTS."
 40  FOR S = 1 TO 4
 50  READ N$,A,B,C
 60  LET T = A + B + C
 70  PRINT "========================================"
 80  PRINT "NAME:   "N$
 90  PRINT "TOTAL:  "T,"AVERAGE: "T / 3
100  NEXT S
110  END
500  DATA  JO,9,8,10,MARIEN,9,10,10
510  DATA  ERNIE,5,7,9,ALICE,10,3,5
```

Run the program.

Each time the Apple travels through the S loop and reads values for N$, A, B and C, the pointer in the DATA statement moves as illustrated with the ⬇ at the right:

Loop with S = 1:

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
          ⬇
```

Loop with S = 2:

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
                        ⬇
```

Loop with S = 3:

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
                        ⬇
501 DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

Loop with S = 4:

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
                                ⬇
501 DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

When the DATA pointer reaches the last item in a DATA statement the Apple searches through the program for another DATA statement and continues from there. Data may be added to this program by adding DATA statements after line 510 and changing the loop number in line 40.

## READ, DATA and RESTORE

It is possible to use the same DATA more than once in a program. A RESTORE statement is used to move the DATA pointer back to the first item in the first DATA statement.

Suppose we want to have the students' names and test scores for each test printed after the averages are printed. We need to RESTORE the DATA pointer before DATA is READ.

Add the following lines to SAMPLE II-5 on page II-5:

```
5  SPEED= 150
110  PRINT
120  PRINT "TEST SCORES"
130  FOR X = 1 TO 4
140  READ N$,A,B,C
150  PRINT "NAME:  "N$
160  PRINT "A="A,"B="B,"C="C
170  PRINT
180  NEXT X
190  SPEED= 255
200  END
```

Run the program.

What message do you get? _____

Add this line: 105 RESTORE
RUN

Location of DATA pointer after loop 5 is finished:

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
501 DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

At line 105, the RESTORE statement moves the pointer back to the first item in the first DATA statement in the program.

```
500 DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
501 DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

Type: SAVE POINTS

## READ DATA with Flags

Using READ within a FOR NEXT loop requires counting the DATA items to prevent the "OUT OF DATA" error message. Another method of using READ is checking for an end of DATA item or "flag". Type in the following program or LOAD SAMPLE II-7 from the intermediate BASIC disk:

```
]NEW
10  HOME
20  PRINT "THIS IS A CHECKBOOK BALANCING PROGRAM"
30  PRINT "<><><><><><><><><><><><><><><><><><><><>"
40  REM  *** BALANCE ***
50  READ B
60  PRINT "BEGINNING BALANCE = $"B
70  REM  *** DEPOSITS ***

80  READ D
90  IF D = 0 THEN 130
100  LET B = B + D
110  GOTO 80

120  REM  *** CHECKS ***

130  READ C
140  IF C = 0 THEN 170
150  LET B = B - C
160  GOTO 130

170  PRINT "ENDING BALANCE = $"B
180  END

500  DATA  593.83
510  DATA  278.04,12,0

520  DATA  9,41.65,7,50,12.02,57.77,11.26,0
```

Line 50 takes the first item in the first DATA statement as the value for B, the beginning balance.

The Apple will continue to loop from line 80-110 and add data to the balance until it reads a value of 0 in a DATA statement.

The Apple will continue to loop from line 130-160 and subtract data from the balance until it reads a value of 0 in the DATA statement.

Line 510 ends in a 0 to "flag" the end of the deposits.

Line 520 ends in a 0 to "flag" the end of the checks.

Run the program.

What is the ending balance? _____


Change line 510 to contain more deposits by typing:

```
510 DATA  278.04, 12, 300, 50, 0
RUN
```

Is the ending balance $350 more? _____

## Variations of the LIST Command

LOAD SAMPLE II-7 from page II-7.

| Type the following. | This will happen |
|---|---|
| LIST<br><br>(Holding the CTRL key down while typing S will temporarily stop a program listing. Holding down the CTRL key while typing S again will resume the listing.) | (The entire program will be listed.)<br>`10 HOME`<br>`20 PRINT "THIS IS A CHECKBOOK BALANCING PROGRAM"`<br>`30 PRINT "<><><><><><><><><><><><><><><><><><><><>"`<br>`40 REM *** BALANCE ***`<br>`50 READ B`<br>`60 PRINT "BEGINNING BALANCE = $" B`<br>`70 REM *** DEPOSITS ***`<br>`80 READ D`<br>`90 IF D = 0 THEN 130`<br>`100 LET B = B + D`<br>`110 GOTO 80`<br>`120 REM *** CHECKS ***`<br>`130 READ C`<br>`140 IF C = 0 THEN 170`<br>`150 LET B = B - C`<br>`160 GOTO 130`<br>`170 PRINT "ENDING BALANCE = $"B`<br>`180 END`<br>`500 DATA 593.83`<br>`510 DATA 278.04,12,0`<br>`520 DATA 9,41.65,7,50,12.02,57.77,11.26,0` |
| LIST, 30 | (All lines up to and including 30 will be listed.)<br>`10 HOME`<br>`20 PRINT "THIS IS A CHECKBOOK BALANCING PROGRAM"`<br>`30 PRINT "<><><><><><><><><><><><><><><><><><><><>"` |
| LIST 500, | (All lines from 500 on will be listed.)<br>`500 DATA 593.83`<br>`510 DATA 278.04,12,0`<br>`520 DATA 9,41.65,7,50,12.02,57.77,11.26,0` |
| LIST 100 | (Only line 100 will be listed.)<br>`100 LET B = B + D` |
| LIST 40, 60<br><br>or<br><br>LIST 40-60 | (All lines from 40 through 60 will be listed.)<br>`40 REM *** BALANCE ***`<br>`50 READ B`<br>`60 PRINT "BEGINNING BALANCE = $" B` |

## Activities

### Program 1: Nested Loops

1. Write a program using nested loops that will produce the following output:

```
XXXXX
X   X
X   X
X   X
X   X
XXXXX

XXXXX
X   X
X   X
X   X
X   X
XXXXX

XXXXX
X   X
X   X
X   X
X   X
XXXXX
```

2. Modify the above program to produce the following output (HINT: Use commas in the PRINT statements.):

```
XXXXX          XXXXX          XXXXX
X   X          X   X          X   X
X   X          X   X          X   X
X   X          X   X          X   X
X   X          X   X          X   X
XXXXX          XXXXX          XXXXX
```

## Activities

### Program 2: READ, DATA

1. LOAD PROGRAM II-2 from the Intermediate BASIC disk or type in the following:

```
]NEW


 10  HOME
 20  FOR T = 1 TO 3
 30  READ R
 40  LET P = 1
 50  FOR N = 1 TO R
 60  LET P = P * N
 70  NEXT N
 80  PRINT "THE PRODUCT OF THE NUMBERS"
 90  PRINT "FROM 1 TO "R" IS "P"."
100  PRINT
110  NEXT T
120  DATA  5,8,12
130  END
```

2. Run the program.

3. Change lines to produce the sum of the numbers rather than the product. Record the lines you changed in the space below:

4. Change a line to produce the sum of the following numbers:

   1 to 10, 1 to 100, 1 to 50 and 1 to 500

   Record the line you changed in the space below:

5. Run the program and record the results in the table below:

| Numbers | Sum |
|---|---|
| 1 to 10 | |
| 1 to 100 | |
| 1 to 50 | |
| 1 to 500 | |

6. Predict the following sums. Then change lines and run the program to record the actual sums.

| Numbers | Sum | |
|---|---|---|
| | Prediction | Actual |
| 1 to 1000 | | |
| 1 to 10000 | | |

## Activities

### Program 3: Reusing an Old Program by Changing DATA

1. LOAD POINTS program from page II-6. Change the DATA in the program to average and print scores on three tests for the following students:

| | Test #1 | Test #2 | Test #3 |
|---|---|---|---|
| HILDA | 7 | 12 | 15 |
| BART | 13 | 7 | 14 |
| JENNY | 12 | 12 | 15 |
| ARTHUR | 15 | 15 | 15 |
| MARTHA | 14 | 10 | 15 |

2. Record the lines you changed in the space below:

### Program 4: READ, DATA AND RESTORE

1. LOAD SAMPLE II-7 from page II-7. Add lines to make the program RESTORE, READ, and PRINT the individual deposits and checks after it prints the ending balance. Record the lines you added in the space below:

After adding and changing lines, type: SAVE CHECKBOOK

2. Change lines 500, 510 and 520 to reflect your checkbook entries and run the program to balance your checkbook.

## Program 5: Using DATA Flags

1. LOAD PROGRAM II-5 from the Intermediate BASIC disk or type in the following:

```
]NEW

10  HOME
20  LET L$ = "========================================"
30  PRINT "*** ALLEY CATS BOWLING  TEAM ***"
40  PRINT L$
50  FOR P = 1 TO 2
60  LET GT = 0: LET NG = 0
70  READ N$
80  PRINT "PLAYER:  "N$
90  READ G1,G2,G3
100  LET T = G1 + G2 + G3
110  LET GT = GT + T
120  LET NG = NG + 3
130  PRINT "SCORES="G1,G2,G3
140  PRINT "TOTAL= "T,"AVERAGE= " T / 3
150  READ F
160  IF F =  - 1 THEN 90
170  PRINT "GRAND TOTAL = "GT
180  PRINT "NUMBER OF GAMES = "NG
190  PRINT "OVERALL AVERAGE = "GT / NG
200  PRINT L$
210  NEXT P
300  END
1000  DATA  MARY
1001  DATA  150,148,219
1002  DATA  -1
1003  DATA  155,145,165
1004  DATA  -2
2000  DATA  JO
2001  DATA  170,130,100
2002  DATA  -1
2003  DATA  140,160,160
2004  DATA  -2
```

2. Run the program.

3. Add lines of DATA to include the following scores:

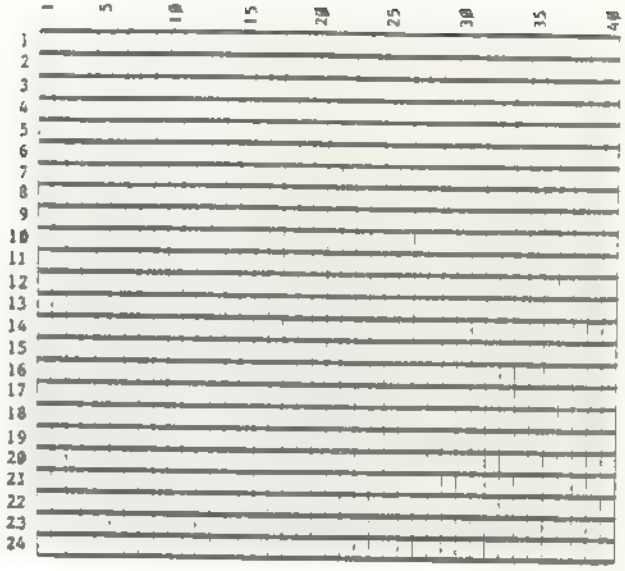
| | |
|---|---|
| MARY: | 188, 200, 175 |
| JO: | 155, 130, 177 |

4. Change the program to include two more players with the following scores:

| | Week 1 | Week 2 | Week 3 |
|---|---|---|---|
| JAN: | 180, 165, 140 | 133, 144, 122 | 150, 140, 150 |
| DOUG: | 179, 164, 139 | 134, 145, 150 | 201, 180, 300 |

## The Text Screen

A diagram of the Apple text screen is shown at the right.




Up to 24 lines may be printed on the text screen. Line 1 is at the top of the screen and line 24 is at the bottom. Printing more than 24 lines will result in scrolling. The top line of the screen will disappear as all lines move up (scroll) one line to make room for the new line at the bottom of the screen.

Each line of print is 40 characters or columns wide. Column 1 is at the far left of the screen. Column 40 is at the far right.

TAB, HTAB and VTAB are three Applesoft BASIC commands to format printed material on the text screen.

---

There are 40 columns across the Apple screen for printing text. To print the word "MICROCOMPUTER" starting at the 5th column and the word APPLE II starting in the 20th column, type in the following program or LOAD SAMPLE III-1 from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "12345678901234567890123456789012345678901234567890"
30  PRINT "    MICROCOMPUTER";"  APPLE II"
40  END
```

Leave 4 spaces between the quote and the word MICROCOMPUTER.

Leave 2 spaces between the quote and the word APPLE.

Run the program.

---

The output should look like this:

```
12345678901234567890123456789012345678901234567890

    MICROCOMPUTER  APPLE II

]
```

## TAB

Having to count and type spaces between quotes and words, in order to get information printed in special columns, could become a very tedious task.

Change this line in SAMPLE III-1 on page III-1:

```
30 PRINT  TAB (5) "MICROCOMPUTER"  TAB (20) "APPLE II"
```

Don't leave any space between the quote and the word.

Run the program.

---

The output should look like this:

```
12345678901234567890123456789012345678901

    MICROCOMPUTER  APPLE II

]
```

---

TAB causes printing to begin at the column number named within parentheses. TAB must be used in a PRINT statement. The number enclosed in parentheses may be a number from 0 to 255. The Apple does not check to see if there is enough room to print at that column. TAB tells the computer where to begin printing. If there is not enough space on the screen to print the information, printing continues on the next line.

Change this line in SAMPLE III-1 on page III-1:

```
30 PRINT  TAB (35) "MICROCOMPUTER"
```

Run the program.

---

The output should look like this:

```
12345678901234567890123456789012345678901

                                   MICRO

OMPUTER

]
```

---

TAB cannot be used to move the printing backwards. If this is attempted the TAB is ignored. Change this line in SAMPLE III-1 on page III-1:

```
30 PRINT  TAB (20) "MICROCOMPUTER"  TAB (30) "APPLE II"
```

Run the program.

## HTAB

The HTAB command works much the same as the TAB function, except it is not used within a PRINT statement. The following programs do the same thing. Type and run each program below or LOAD and run each sample from the Intermediate BASIC disk:

SAMPLE III-2A

```
]NEW

10  HOME
20  SPEED= 150
30  FOR X = 1 TO 40
40  PRINT  TAB( X)"*"
50  NEXT X
60  SPEED= 255
70  END
```

SAMPLE III-2B

```
]NEW

10  HOME
20  SPEED= 150
30  FOR X = 1 TO 40
40  HTAB X
50  PRINT "*"
60  NEXT X
70  SPEED= 255
80  END
```

In SAMPLE III-2A, TAB is within the PRINT statement. The column number is enclosed within parentheses.

In SAMPLE III-2B, HTAB is a separate line of the program. The HTAB number is not enclosed within parentheses.

---

HTAB will print information in the column named ignoring what may already be printed there. Type in the following or LOAD SAMPLE III-2C from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  SPEED= 0
30  HTAB 10
40  PRINT "MICROCOMPUTER";
50  HTAB 10
60  PRINT "APPLE"
70  SPEED= 255
80  END
```

Run the program.

Change this line by leaving off the semicolon:

```
40 PRINT "MICROCOMPUTER"
```

Run the program.

## VTAB

There are 24 lines (40 columns wide) on the Apple screen for printing text. The lines are numbered 1 to 24 from top to bottom. A VTAB will cause the Apple to print at the line number following the VTAB command. Only numbers 1 through 24 may be used with VTAB. Type in the following or LOAD SAMPLE III-3A from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  SPEED= 100
30  VTAB 1
40  PRINT "THIS IS ON LINE NUMBER 1."
50  VTAB 16
60  PRINT "IS FUN"
70  VTAB 10
80  PRINT "PROGRAMMING"
90  VTAB 13
100  PRINT "THE APPLE"
110  VTAB 20
120  PRINT "AND EASY."
130  SPEED= 255
140  END
```

Before running the program, predict what the screen output will look like. Record your prediction in the space below.

Run the program.

---

VTAB will print on the line named, ignoring what may already be printed there. Change the following line in the above program:

```
90 VTAB 10
```

Run the program.

---

VTAB and HTAB may be combined to place printing both horizontally and vertically on the screen. Type in the following or LOAD SAMPLE III-3B from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  HTAB 13: VTAB 11
30  FLASH:PRINT"I AM THINKING":NORMAL
40  END
```

Run the program.

## Pausing with Loops

A FOR-NEXT loop may be inserted between screens of instructions to slow down the time between the first set of instructions being printed and the screen being cleared for the next display. There is nothing in the body of the loop. It is only included to take up time. Type the following or LOAD SAMPLE III-4 from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  INPUT "WHAT IS YOUR NAME? ";A$
30  PRINT : PRINT "HELLO, " A$ "." : PRINT
40  VTAB 22
50  PRINT "ONE MOMENT PLEASE . . ."
60  FOR P = 1 TO 1500: NEXT P
70  HOME
80  PRINT "ARE YOU ENJOYING BASIC?"
90  INPUT "YES OR NO ";Q$
100  VTAB 22
110  PRINT "LET ME THINK A MINUTE ABOUT THAT . . ."
120  FOR P = 1 TO 1500: NEXT P
130  IF Q$ = "YES" THEN 160
140  IF Q$ = "NO" THEN 180
150  GOTO 90
160  HOME : FLASH : PRINT "I'M GLAD!!!": NORMAL
170  GOTO 190
180  HOME : SPEED= 110: PRINT "I'M SORRY TO HEAR
THAT.": SPEED=255

190  FOR P = 1 TO 1500: NEXT P
200  HOME
210  PRINT "GOOD BYE, " A$ "."
220  END
```

This gets the Apple going around in circles 1500 times to take up time.

Run the program. Answer the questions.

---

In the above example, the program determines the speed at which the user is led through each screen of information. Change the following lines in the above program to give the person running the program more control over the speed at which the user is led through each screen of information:

```
50 PRINT "PRESS ANY KEY TO GO ON...";
60 GET K $
110 PRINT "PRESS ANY KEY TO GO ON...";
120 GET K $
190 PRINT "PRESS ANY KEY TO CONTINUE"; : GET K$
```

Run the program.

## GET

The GET statement introduced at the bottom of the previous page is similar to the INPUT statement. GET makes the Apple wait for the person running the program to press a single key. The value of the key pressed is stored in the variable location following the word GET. If a numeric variable is used with GET (GET A, GET X, etc.) only number keys pressed will be accepted. If a string variable is used with GET (GET A$, GET X$, etc.) any key pressed will be accepted. It is a good idea to use string variables with GET.

Once a single key is pressed, the Apple goes on with the next statement of the program. The RETURN key does not have to be pressed as with the INPUT statement. Type in the following or LOAD SAMPLE III-5 from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "PRESS ANY KEY ON THE KEYBOARD";
30  GET T$
40  PRINT : PRINT
50  PRINT "YOU PRESSED:   " T$
60  VTAB 12
70  PRINT "AGAIN?  Y OR N ";
80  GET A$
90  IF A$ = "Y" THEN 10
100  HOME : PRINT "SO LONG."
110  END
```

Run the program.

Answer "Y" to the "AGAIN?" question and try pressing different number, letter and character keys.

---

Change the following lines to use number variables instead of string variables in the GET statement:

```
30 GET T
50 PRINT "YOU PRESSED:  " T
```

Run the program and press number keys only. Try a letter key.

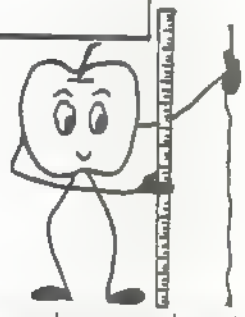
What happens? _____

## LEN and STR$

The LEN function will determine the length of a string variable value. The Apple will count all spaces, punctuation and characters in calculating the length of a string. Type in the following or LOAD SAMPLE III-6 from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "TYPE ANYTHING YOU WISH."
30  INPUT S$
40  PRINT : PRINT
50  PRINT "YOU TYPED:  ": PRINT S$:PRINT
60  LET L =  LEN (S$)
70  PRINT "THERE ARE "L" CHARACTERS AND SPACES."
80  PRINT : PRINT "AGAIN?  Y OR N ";: GET K$
90  IF K$ = "Y" THEN 10
100  HOME : PRINT "FAREWELL!"
110  END
```



Run the program.

Type a single word the first time.

Answer "Y" to the "AGAIN?" question and try longer messages.

*(CAUTION: The maximum length of a string is 255. Try something longer when running the program. What is the error message? _____)

---

To determine the length of a numeric INPUT, the numeric value must first be changed to a string value. LEN only works for string values. STR$ will change a numeric value to a string value. Change and add the following lines in the above program:

```
20 PRINT "TYPE A NUMBER."
30 INPUT N
35 LET S$ = STR$ (N)
```

Run the program.

Try the following numbers and record the number of characters in each:

| | | |
|---|---|---|
| 37 ___ characters | .2815 ___ characters | -12345 ___ characters |
| 37.8 ___ characters | 0.2815 ___ characters | +12345 ___ characters |

## LEFT$ and RIGHT$

The LEFT$ function will cause the Apple to look at the left part of a string. Type the following or LOAD SAMPLE III-7 from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "TYPE A WORD:  ";
30  INPUT W$
40  PRINT : PRINT "YOU TYPED:  " W$
50  PRINT "IT HAS " LEN (W$)" CHARACTERS."
60  PRINT "THE FIRST LETTER IS " LEFT$(W$,1)"."
70  VTAB 12: INPUT "AGAIN? ";A$
80  IF  LEFT$ (A$,1) = "Y" THEN 10
90  HOME
100  PRINT "SEE YA!"
110  END
```

Run the program.
Type any word you wish.
Answer "YES" to the "AGAIN?" question.
Type any word you wish.
Answer "YES I'D LOVE TO" to the "AGAIN?" question.
(Line 8Ø only checks the first character of your answer in A$. Any answer beginning with "Y" is understood to mean you wish to type another word.)

---

Change the following line in the above program:

```
6Ø PRINT "THE FIRST TWO LETTERS ARE " LEFT$ (W$, 2)"."
```

Run the program.

---

Change a line to make the program print the first 3 characters of any word typed. Record the line you changed in the space below:

Run the program.
What happens when you type a word with only 2 letters?

---

Change the following line in the above program:

```
6Ø PRINT "THE LAST LETTER IS  " RIGHT$ (W$, 1)"."
```

Run the program.

---

Change a line to make the program print the last two letters of any word that is typed. Record the line you changed in the space below:

Run the program.

## MID$

The MID$ function is a way to look at the middle parts of strings. Type the following or LOAD SAMPLE III-8A from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "TYPE A WORD WITH AT LEAST 8 CHARACTERS."
30  INPUT W$
40  PRINT : PRINT
50  IF  LEN (W$) < 8 THEN 20
60  PRINT "YOU TYPED:   " W$
70  PRINT "IT HAS " LEN (W$)" CHARACTERS."
80  PRINT : PRINT
90  PRINT "THE PART OF YOUR WORD"
100  PRINT "BEGINNING WITH THE 2ND LETTER"
110  PRINT "AND PRINTING 4 LETTERS IS ";
120  PRINT  MID$ (W$,2,4) "."
130  VTAB 20
140  INPUT "AGAIN? ";A$
150  IF  LEFT$ (A$,1) = "Y" THEN 10
160  HOME
170  END
```

Run the program.

---

MID$ may be used to look through a string, one character at a time, in search of some special character. Type the following or LOAD SAMPLE III-8B from the Intermediate BASIC disk:

```
]NEW

10  HOME
20  PRINT "TYPE A NUMBER WITH A DECIMAL POINT."
30  INPUT N$
40  PRINT : PRINT
50  LET L =  LEN (N$)
60  FOR A = 1 TO L
70  LET M$ =  MID$ (N$,A,1)
80  IF M$ = "." THEN 120
90  NEXT A
100  PRINT "THERE IS NO DECIMAL POINT IN "N$ " "
110  GOTO 150
120  PRINT "THE DECIMAL POINT IS CHARACTER "A "."
130  PRINT "THERE ARE "A - 1" CHARACTERS BEFORE "
140  PRINT "AND "L - A" AFTER THE DECIMAL POINT."
150  INPUT "AGAIN? "; A$
160  IF LEFT$(A$,1) = "Y" THEN 10
170  HOME
180  END
```

Run the program. What happens when each of the following numbers are used?

4.7890  .9  0.9  -3.768  +5.6  298

## Activities

### Program 1: TAB

Write and run a program using TAB to print your name, rank and phone number centered on the T.V. screen as shown below:

```
MARCIA HORN
ROOKIE
633-9110 EXT. 195
```

### Program 2: HTAB and VTAB

1. LOAD PROGRAM III-2 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME
20  PRINT "PLEASE TYPE YOUR FIRST NAME."
30  INPUT F$
40  PRINT "PLEASE TYPE YOUR LAST NAME."
50  INPUT L$
60  HOME
70  PRINT L$
80  PRINT F$
90  END
```

2. Run the program.

3. Change and add lines using HTAB and VTAB to print the first name somewhere at the bottom center of the screen and the last name somewhere at the upper center of the screen. Record the lines you add and change in the space below:

4. Run the program.

## Activities

### Program 3: FOR-NEXT Pausing Loop

Write and run a program using the FOR-NEXT pausing loop that will:

| Make the Apple do the following. | BASIC statement to do it. |
|---|---|
| Clear the screen. | |
| Print "GET READY". | |
| Pause. | |
| Clear the screen. | |
| Print "GET SET". | |
| Pause. | |
| Clear the screen. | |
| Flash "GO!" on the screen. | |
| Pause. | |
| Clear the screen. | |
| Print a good-bye message. | |

### Program 4: GET

1. Change and add lines in the above program that will give the person running the program control over the speed at which screens are changed. Use GET and VTAB lines. Record the lines you change in the space below:

2. Run the program.

## Activities

### Program 5: GET

Write a program using PRINT, INPUT, GET and IF-THEN to produce the following output. (Underlined words are typed in at the keyboard by the person running the program.)

The output will look like the following when answering "Y":

Screen 1:

```
]RUN
WHAT IS YOUR NAME? DAVID

HI, DAVID.

PRESS ANY KEY TO CONTINUE
```

Screen 2:

```
HOPE YOU ARE HAVING FUN WITH APPLESOFT
INTERMEDIATE BASIC.

DO YOU UNDERSTAND THE GET STATEMENT?

PLEASE TYPE Y FOR YES AND N FOR NO: Y

SUPER
```

The output will look like the following when answering anything other than "Y or N" and then answering "N":

Screen 1:

```
]RUN
WHAT IS YOUR NAME? MAGGIE

HI, MAGGIE.

PRESS ANY KEY TO CONTINUE
```

Screen 2:

```
HOPE YOU ARE HAVING FUN WITH APPLESOFT
INTERMEDIATE BASIC.

DO YOU UNDERSTAND THE GET STATEMENT?

PLEASE TYPE Y FOR YES AND N FOR NO: P

PLEASE TYPE Y FOR YES AND N FOR NO: N

OH MY.  TIME TO BONE UP.
```

## Activities

### Program 6: LEN

1. LOAD PROGRAM III-6 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME
20  VTAB 8: HTAB 5
30  PRINT "THIS IS A QUIZ ABOUT APPLES."
40  PRINT : INPUT "WHAT IS YOUR NAME? ";N$
50  VTAB 16: PRINT "PRESS ANY KEY";: GET K$
60  FOR Q = 1 TO 3
70  HOME : VTAB 5
80  READ Q$,CA$
90  PRINT Q$
100  PRINT : INPUT A$
110  IF CA$ = A$ THEN 190

160  PRINT : PRINT "SORRY, "N$", THE ANSWER IS"
170  PRINT CA$"."
180  GOTO 200
190  PRINT : PRINT "CORRECT, "N$"!!!"
200  VTAB 23: PRINT "PRESS ANY KEY";: GET K$
210  NEXT Q
220  HOME : PRINT "SO LONG, " N$ "."
230  END

400  DATA  WHAT IS THE APPLE'S LANGUAGE CALLED?,BASIC
410  DATA  PROGRAMS ARE STORED PERMANENTLY ON -?-.,DISK
420  DATA  THERE ARE --?-- SECTORS ON A DISK.,560
```

2. Run the program.

3. Add lines between 11Ø and 16Ø and use LEN to give the person running the program the following hint and a second chance before telling them the correct answer:

Sample Screen:

```
WHAT IS THE APPLE'S LANGUAGE CALLED?

?FORTRAN

NO, MARCIA.
HINT-THE ANSWER HAS 5 LETTERS.
TRY AGAIN ? COBOL

SORRY, MARCIA, THE ANSWER IS
BASIC.
```

4. SAVE APPPLE QUIZ

## Activities

### Program 7: LEFT$

1. LOAD APPLE QUIZ from Activity Program 6 on page III-13. Add lines using LEFT$ to give the person running the program a second hint and a third try before telling them the correct answer.

Sample Screen:

```
WHAT IS THE APPLE'S LANGUAGE CALLED?

?FORTRAN

NO, HEIDI.
HINT-THE ANSWER HAS 5 LETTERS.
TRY AGAIN ? COBOL

NO.  THE FIRST LETTER IS B.
TRY AGAIN ? BRAIN

SORRY, HEIDI, THE ANSWER IS
BASIC.
```

2. Run the program and answer questions incorrectly.

3. Run the program again and answer questions correctly.

4. Optional: Change the DATA statements to your own questions and correct answers. Change line 60 if you want more than 3 questions. SAVE (give your quiz a name).

---

### Program 8: RIGHT$

1. LOAD PROGRAM III-8 from the Intermediate BASIC disk or type the following:

```
]NEW

 10  HOME
 20  INPUT "TYPE A WORD:  "; W$
 30  SPEED= 150
 40  LET L =  LEN (W$)
 50  FOR A = 1 TO L
 60  PRINT  LEFT$ (W$,A)
 70  NEXT A
 80  SPEED= 255
 90  END
```

2. Run the program.

3. Change a line to print the word typed in parts starting with the right of the word as below:

Sample Run:

```
TYPE A WORD:  CAT
T
AT
CAT
```

## Activities

### Program 9: MID$

1. LOAD SAMPLE III-8B from page III-8.

2. Change the program to search through any word typed in and look for the letter X. The output should look like the following:

If an X is in the word:

```
TYPE A WORD:
?MAXIMUM

THE X IS CHARACTER 3.
THERE ARE 2 CHARACTERS BEFORE
AND 4 AFTER THE X.
AGAIN?
```

If there is no X in the word:

```
TYPE A WORD:
?MINIMUM

THERE IS NO X IN MINIMUM.
AGAIN?
```

If an X is the first letter:

```
TYPE A WORD:
?X-RAY

THE X IS CHARACTER 1.
THERE ARE 0 CHARACTERS BEFORE
AND 4 AFTER THE X.
AGAIN?
```

If an X is the last letter:

```
TYPE A WORD:
?AX

THE X IS CHARACTER 2.
THERE ARE 1 CHARACTERS BEFORE
AND 0 AFTER THE X.
AGAIN?
```

3. CHALLENGE!!! Add and change lines using LEFT$ and RIGHT$ to print the letters that come before and after the first X found in a word.

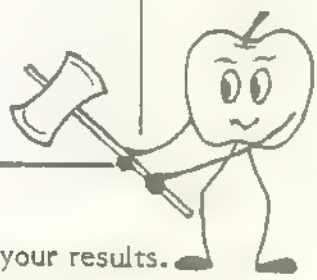
## INT (the INTeger Function)

Results of computation in Applesoft BASIC are given as decimals. In division, lengthy decimal remainders are quite common. The program application may only require a whole number part or a decimal rounded to a set number of places. INT is an Applesoft function that will help manage these very precise decimal answers. LOAD SAMPLE IV-1 from the Intermediate BASIC disk or type the following.

```
]NEW

 10  HOME
 20  FOR X = 1 TO 5
 30  INPUT "TYPE A NUMBER:  "; N
 40  INPUT "DIVIDED BY:  "; D
 50  PRINT N" DIVIDED BY "D" = "N / D
 60  PRINT "INT(N/D) = " INT (N / D)
 70  PRINT : PRINT
 80  PRINT "PRESS ANY KEY TO GO ON ";: GET K$
 90  HOME
 100  NEXT X
 110  PRINT "DO YOU UNDERSTAND INT?"
 120  PRINT "IF NOT, TYPE RUN"
 130  PRINT "AND TRY MORE NUMBERS."

 140  END
```



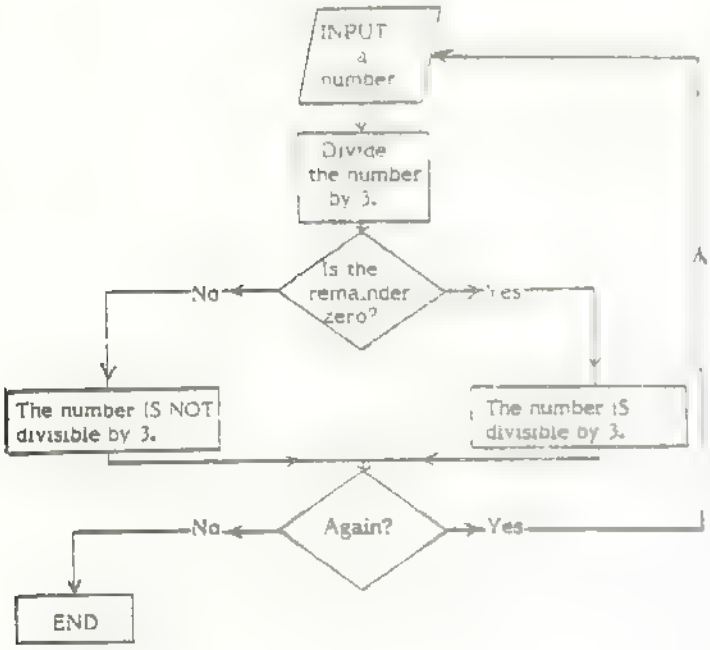
Run the program. Answer with numbers from the table below. Record your results.

| NUMBER (N) | DIVIDED BY (D) | N/D | INT(N/D) |
|---|---|---|---|
| 36 | 9 | | |
| 36 | 5 | | |
| 89 | 3 | | |
| 7 | 8 | | |
| -13 | 4 | | |

INT tells the Apple to evaluate the expression within parentheses and then compute the greatest integer less than or equal to the value of the expression. INT rounds down and drops the decimal part. The integer function returns the integer that is just to the left of the value on the number line.

## INT to Check Divisibility

INT is a useful function in mathematics as a check for divisibility and finding factors. To determine if a number is divisible by 3, the number is divided by 3 and the remainder is examined. If the remainder is zero then the number is divisible by 3. The following flow chart and program illustrate the use of INT in a check for divisibility. LOAD SAMPLE IV-2 from the Intermediate BASIC disk or type the following:




```
NEW

10  HOME
20  INPUT "TYPE A NUMBER:  "; N

30  LET Q = N / 3

40  IF Q =  INT (Q) THEN 70

50  PRINT N" IS NOT DIVISIBLE BY 3."
60  GOTO 80

70  PRINT N" IS DIVISIBLE BY 3."

80  PRINT
90  INPUT "AGAIN - Y OR N ? ";A$
100  IF  LEFT$ (A$,1) = "Y" THEN 10

110  HOME
120  END
```

---

Run the program. Circle the numbers below that have 3 as a factor (are divisible by 3):

8 15 421 822 943 21235 37037034

---

Change the lines in the program to check if numbers are divisible by 8. Record the lines you changed in the space below:

Run the program. Circle the numbers below that are divisible by 8 (have 8 as a factor):

16 28 184 1000 188 1888 18888

## INT to Round Decimals

Chopping off the decimal part may not always be desirable...especially in work with dollar amounts, averages and percents. INT is a useful function in rounding decimal numbers. If a number is to be rounded to the nearest hundredth (penny), the Apple must go through the following steps:

| | | Number to be rounded = 11.5871 |
|---|---|---|
| 1. | Add half of the rounding unit to the number (½ of .01 = .005). | 11.5871<br>+.005<br>11.5921 |
| 2. | Multiply by 100. | x100<br>1159.21 |
| 3. | Determine the greatest integer. | INT (1159.21) = 1159 |
| 4. | Divide the number by 100. | 1159/100 = 11.59 |

LOAD SAMPLE IV-3 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME
20  INPUT "TYPE A NUMBER:  "; N
30  LET R = N + .005: PRINT R
40  LET R = R * 100: PRINT R
50  LET R =  INT (R): PRINT R
60  LET R = R / 100
70  PRINT N" ROUNDED TO HUNDREDTHS IS "R "."
80  INPUT "AGAIN - Y OR N ? ";A$
90  IF  LEFT$ (A$,1) = "Y" THEN 10
100  HOME
110  END
```

Run the program. Use the following numbers and record the results below:

| Number | Rounded to hundredths | Rounded to tenths |
|---|---|---|
| 3.143 | | |
| 57.2167 | | |
| .9999 | | |
| 4.6 | | |
| 137.49666 | | |

Change lines in the program to round numbers to the nearest tenth. Record the lines you changed in the space below:

Run the program. Use the numbers used in rounding to hundredths. Record your results in

## RND (the RaNDom Function)

A good part of the appeal of computers in drill and simulations is their ability to provide unpredictable problems and situations. The RND function causes the computer to select a number randomly. The random number can then be used as part of a math problem or signal the computer to take a certain set of steps depending on the number selected. LOAD SAMPLE IV-4 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME
20  FOR X = 1 TO 20
30  LET R =  RND (1)
40  PRINT R
50  NEXT X
60  END
```

Run the program. Record 3 of the random numbers below:

_____ _____ _____

The Apple uses scientific notation to print numbers with more than nine digits. In the number 3.53799262 E-Ø3, the "E-Ø3" means move the decimal point to the left 3 places:

3.53799262 E-Ø3 = Ø.ØØ353799262

In the number 1.23456789 E+12, the "E+12" means move the decimal point to the right 12 places:

1.23456789 E+12 = 1234567890ØØØ

Run the program again. Record the last random number selected:

_____

---

Change line 3Ø to: 3Ø LET R = RND (Ø)

Run the program at least two times.

What do you notice? _____

---

Change line 3Ø to: 3Ø LET R = RND (-3)

Run the program at least two times.

What do you notice? _____

---

Change line 3Ø to: 3Ø LET R = RND (4)

Run the program at least two times.

What do you notice? _____

## Better RND Numbers

The RND function causes the Apple to randomly select a decimal number greater than or equal to zero but less than one. To make certain that different random numbers are selected each time the program is run, use a number greater than Ø with the RND function. A common practice is to use RND(1) to generate random numbers.

The random numbers selected by the RND function are a bit awkward in their decimal form. Whole numbers may be easier to use. Some additional steps are necessary to make whole numbers of the random decimals. LOAD SAMPLE IV-5 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME
20  PRINT  TAB( 5)"R" TAB( 20)"10 *R" TAB( 35)"INT"
30  PRINT "----------" TAB( 17)"----------" TAB( 34)"-----"
40  FOR X = 1 TO 10
50  LET R =  RND (1)
60  PRINT R TAB( 17)10 * R TAB( 36) INT (10 * R)
70  PRINT
80  NEXT X
90  END
```

---

Lines 5Ø and 6Ø in the program above show the steps involved in changing a random decimal number to a whole number.

| | | |
|---|---|---|
| A. | Select a random number: | R = RND (1) = .712763244 |
| B. | Multiply the random number by 1Ø: | 1Ø * R = 7.12763244 |
| C. | Chop off the decimal part: | INT (1Ø * R) = INT (7.12763244) = 7 |

---

Run the program at least two times.

What is the largest number listed under INT? _____

What is the smallest number listed under INT? _____

---

Change these lines: 2Ø PRINT TAB(5) "R" TAB (2Ø) "1ØØ * R "TAB (35) "INT"

6Ø PRINT R TAB (17) 1ØØ * R TAB (36) INT (1ØØ *R)

Run the program at least two times.

What is the largest number listed under INT? _____
What is the smallest number listed under INT? _____

## RND

The steps involved in selecting and preparing a whole number randomly may be combined in one line of the program. LOAD SAMPLE IV-6 from the Intermediate BASIC disk or type the following:

*Line 40 is the random number generator.

```
]NEW

10  HOME
20  SPEED= 150
30  FOR X = 1 TO 20

40  LET R =  INT (100 *  RND (1))

50  PRINT "I PICKED:  " R
60  NEXT X
70  SPEED= 255
80  END
```

Run the program.

The smallest number you can get is 0 and the largest is 99. To get the numbers in the range of 1 to 100, change the following line:

40 LET R = INT ( 100 * RND (1) ) + 1

Run the program at least two times.

In general, the random number line may be interpreted as follows:

LET R = INT ( 6 * RND (1) ) + 1

.... will produce 6 random numbers starting with 1....
1, 2, 3, 4, 5, 6

LET R = INT ( 11 * RND (1) ) + 2

.... will produce 11 random numbers starting with 2 ....

2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

Change line 40 to produce random numbers in the range of 15 to 40 (15, 16, 17,... , 40). Record the line you changed in the space below:

Run the program at least two times.

## RND to Flip a Coin and Drill in Addition

LOAD SAMPLE IV-7A from the Intermediate BASIC disk or type the following:

```
]NEW
10  HOME
20  LET T = 0: LET H = 0
30  FOR F = 1 TO 10
40  PRINT "PRESS ANY KEY TO FLIP
     THE COIN ...";: GET K$
50  LET C =  INT (2 *  RND (1)) + 1
60  IF C = 1 THEN 100
70  LET T = T + 1
80  PRINT "TAILS!"
90  GOTO 120
100  LET H = H + 1
110  PRINT "HEADS!"
120  NEXT F
130  PRINT "NUMBER OF TAILS:  "T
140  PRINT "NUMBER OF HEADS: "H
150  END
```

Run the program.

How many TAILS? _____ How many HEADS? _____

Run the program again.

How many TAILS? _____ How many HEADS? _____

---

LOAD SAMPLE IV-7B from the Intermediate BASIC disk or type the following:

```
]NEW
10  HOME
20  FOR P = 1 TO 4
30  LET X =  INT (20 *  RND (1)) + 1
40  LET Y =  INT (10 *  RND (1)) + 1
50  LET S = X + Y
60  PRINT X" + "Y" = ";
70  INPUT A
80  IF A = S THEN 110
90  PRINT "NOPE - TRY AGAIN."
100  GOTO 60
110  PRINT "CORRECT ! ! !"
120  VTAB 20: PRINT "PRESS ANY KEY TO GO ON ...";: GET K$
130  HOME
140  NEXT P
150  PRINT : PRINT "THAT IS ALL FOR NOW!!!"
160  PRINT : PRINT "TAKE A COFFEE BREAK!!!"
170  END
```

Run the program. Answer problems correctly and incorrectly. Change lines in the program to print 8 multiplication problems with X in the range of 1 to 9 and Y in the range of 1 to 12. Record the lines you changed in the space below:

Run the program. SAVE MULTIPLICATION

## GOSUB...RETURN

There are occasions when a series of statements within a program needs to be used several times. Rounding the results of computations, printing the "PRESS ANY KEY TO CONTINUE" message to clear the screen, and checking input are examples of statements that may be needed several times in one program. GOSUB sends the Apple to a line of the program. The Apple continues to work from that line number and on (the subroutine). When a RETURN statement is reached the Apple returns to the line following the GOSUB statement. LOAD SAMPLE IV-8 from the Intermediate BASIC disk or type the following:

```
]NEW

10  HOME : VTAB 10
20  PRINT "TYPE IN ANY NUMBER."
30  PRINT "I'LL DIVIDE IT BY 19 AND"
40  PRINT "ROUND THE ANSWER TO THE"
50  PRINT "NEAREST HUNDREDTH."
60  INPUT N
70  LET X = N / 19
80  GOSUB 300
90  PRINT : PRINT N" DIVIDED BY 19 = "N / 19
100  PRINT N / 19" ROUNDED IS "X "."
110  GOSUB 400
120  PRINT "TYPE A DOLLAR AND CENTS AMOUNT."
130  PRINT "I'LL CALCULATE THE SALES TAX"
140  PRINT "AND TOTAL COST.  DON'T USE A $"
150  INPUT "IN YOUR MONEY AMOUNT. ";D
160  LET X = .04 * D
170  GOSUB 300
180  PRINT : PRINT "TAX = $"X
190  PRINT : PRINT "TOTAL = $"D + X
200  GOSUB 400
210  INPUT "AGAIN - Y OR N ? ";A$
220  IF  LEFT$ (A$,1) = "Y" THEN 10
230  GOTO 500


300  LET X = X + .005
310  LET X = X * 100
320  LET X =  INT (X)
330  LET X = X / 100
340  RETURN


400  VTAB 22
410  PRINT "PRESS ANY KEY TO CONTINUE...";: GET K$
420  HOME : VTAB 10
430  RETURN


500  HOME
510  END
```

Lines 300-340
The subroutine to round X to the nearest hundredth.

Lines 400-430
The subroutine to clear the screen when a key is pressed.

Run the program.

---

Lines 300, 310, 320, and 330 may be shortened to one line: 300 LET X = INT ( (X + .005) * 100)/100

*(A line like 230 should be placed before the subroutine section of a program. Without it a "RETURN WITHOUT GOSUB" error message will be printed when the "AGAIN" question is not answered "Y".)

## ON....GOSUB

The subroutine to be performed may be dependent on the user's input. The ON....GOSUB statement will direct the Apple to the subroutine the user selects. LOAD SAMPLE IV-9 from the Intermediate BASIC disk or type the following:

Lines 30-50
The addition subroutine

Lines 60-80
The subtraction subroutine

Line 90
The multiplication subroutine

Line 100
The division subroutine

Line 110
The clear the screen subroutine

```
]NEW
10  HOME
20  GOTO 130

30  LET S$ = " + "
40  LET A = X + Y
50  RETURN

60  LET S$ = " - "
70  LET X = X + Y: LET A = X - Y
80  RETURN

90  LET A = X * Y: LET S$ = " * " : RETURN

100  LET A = X: LET X = A * Y: LET S$ = " / ": RETURN

110  VTAB 23: PRINT "PRESS ANY KEY TO GO ON";: GET K$
120  HOME : RETURN

130  VTAB 10
140  LET X =  INT (10 *  RND (1)) + 1
150  LET Y =  INT (10 *  RND (1)) + 1
160  PRINT "WHICH OPERATION:": PRINT
170  HTAB 5: PRINT "(1)  ADDITION"
180  HTAB 5: PRINT "(2)  SUBTRACTION"
190  HTAB 5: PRINT "(3)  MULTIPLICATION"
200  HTAB 5: PRINT "(4)  DIVISION"
210  HTAB 5: PRINT "(5)  END"
220  INPUT "TYPE 1, 2, 3, 4 OR 5: ";C
230  IF C < 1 OR C > 5 THEN 220
240  IF C = 5 THEN 350

250  ON C GOSUB 30,60,90,100

260  HOME : VTAB 10
270  PRINT X;S$;Y;" = ";
280  INPUT R
290  IF A = R THEN 320
300  PRINT "NO, THE ANSWER IS:  " A
310  GOTO 330
320  PRINT "THAT IS IT ! ! !"
330  GOSUB 110
340  GOTO 130
350  HOME
360  END
```

Line 250 means:
If C = 1 then GOSUB 30
If C = 2 then GOSUB 60
If C = 3 then GOSUB 90
If C = 4 then GOSUB 100

Run the program and try options 1, 2, 3, 4 and 5.

---

*(A subroutine can be placed anywhere in a program. With the Apple, a program will run faster if the subroutines are placed at the beginning of the program.)

## ON....GOTO

The ON...GOTO statement functions much the same as ON...GOSUB. ON...GOTO sends the computer to a line where program execution continues. BUT there is no returning to the place where control was originally transferred. The following program will count the outcomes of a die being rolled 100 times. LOAD SAMPLE IV-10 from the Intermediate BASIC disk or type the following:

```
]NEW
10  HOME
15  FLASH : VTAB 12: HTAB 15: PRINT
     "PLEASE WAIT.": NORMAL
16  VTAB 13: HTAB 4: PRINT "I AM
     ROLLING THE DIE 100 TIMES."
20  FOR R = 1 TO 100
30  LET D =  INT (6 *  RND (1)) + 1

40  ON D GOTO 50,60,70,80,90,100

50  LET D1 = D1 + 1: GOTO 110
60  LET D2 = D2 + 1: GOTO 110
70  LET D3 = D3 + 1: GOTO 110
80  LET D4 = D4 + 1: GOTO 110
90  LET D5 = D5 + 1: GOTO 110
100  LET D6 = D6 + 1
110  NEXT R
115  HOME : VTAB 9
120  PRINT : PRINT "1 WAS ROLLED "D1" TIMES."
130  PRINT "2 WAS ROLLED "D2" TIMES."
140  PRINT "3 WAS ROLLED "D3" TIMES."
150  PRINT "4 WAS ROLLED "D4" TIMES."
160  PRINT "5 WAS ROLLED "D5" TIMES."
170  PRINT "6 WAS ROLLED "D6" TIMES."

180  END
```

Line 40:

If the random number is 1...the program goes to line 50
if D is 2, then 60...
if D is 3, then 70...
if D is 4, then 80...
if D is 5, then 90...
if D is 6, then 100.

Run the program twice. Record the results in the table below.

| Number | Times in 100 Rolls | Times in 100 Rolls | Times in 1000 Rolls |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |

Change lines to have the computer count for 1000 rolls of the die. Run the program. Record the results in the table above.

---

If the value of the variable in either an ON....GOSUB or an ON....GOTO statement is 0 or greater than the number of lines listed, the ON....GOSUB or ON....GOTO statement is skipped and the computer proceeds with the next line of the program.

```
250 ON C GOSUB 30, 60, 90
260 PRINT "AN APPLE A DAY."
```

If C equals 0 or is greater than 3, line 260 will be executed next.

```
250 ON C GOTO 10, 20, 30, 40
260 PRINT "APPLES ARE HERE TO STAY."
```

If C equals 0 or is greater than 4, line 260 is executed next.

---

If C is less than 0 or greater than 255 an "ILLEGAL QUANTITY ERROR" message will be printed.

## Activities

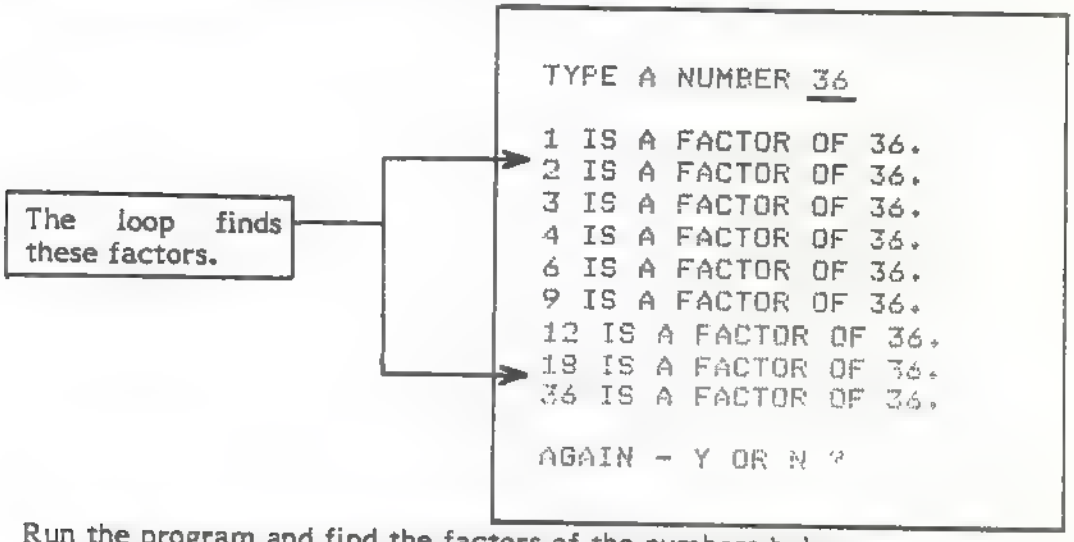
### Program 1: INT for listing factors

1. LOAD SAMPLE IV-2 from page IV-2.

2. Modify the program to print the factors of a number by adding a FOR NEXT loop that will:

   A. Divide the INPUT number by each whole number from 2 through ½ the number (FOR D = 2 TO N/2).
   B. Check the INPUT number to see if it is divisible by each of the whole numbers.
   C. If the INPUT number is not divisible by the whole number, go to the next whole number.
   D. If the INPUT number is divisible by the whole number, then print the whole number (it is a factor).

   The output should look like the following:




3. Run the program and find the factors of the numbers below:

| Number | Factors |
|---|---|
| 6 | |
| 13 | |
| 24 | |
| 60 | |
| 419 | |
| 1000 | |

4. A prime number only has factors of 1 and itself. (5 is prime because its factors are 1 and 5.) Run the program using the numbers listed below. Circle the numbers that are prime.

   7 19 33 37 51 57 1009 1107 1999 2003

5. SAVE FACTOR

## Activities

### Program 2: INT to round numbers

1. LOAD SAMPLE IV-3 from the Intermediate BASIC disk.
2. Modify the program to round a number to the nearest whole number answer (194.73 would be rounded to 195).
3. Record how the Apple rounds each of the following:

   3.6 ____ .001 ____ 589 ____ 74.05 ____ 74.5 ____

### Program 3:

1. LOAD MULTIPLICATION from page IV-7.
2. Modify the program to ask the user the number of problems they would like. Record the lines you add and change in the space below:

3. Change the program to print the correct answer when the problem is answered incorrectly. Record the lines you change in the space below:

4. Add a score-keeping counter to count the number of problems the user answers correctly. Have the number correct printed at the end. Record the lines you add in the space below:

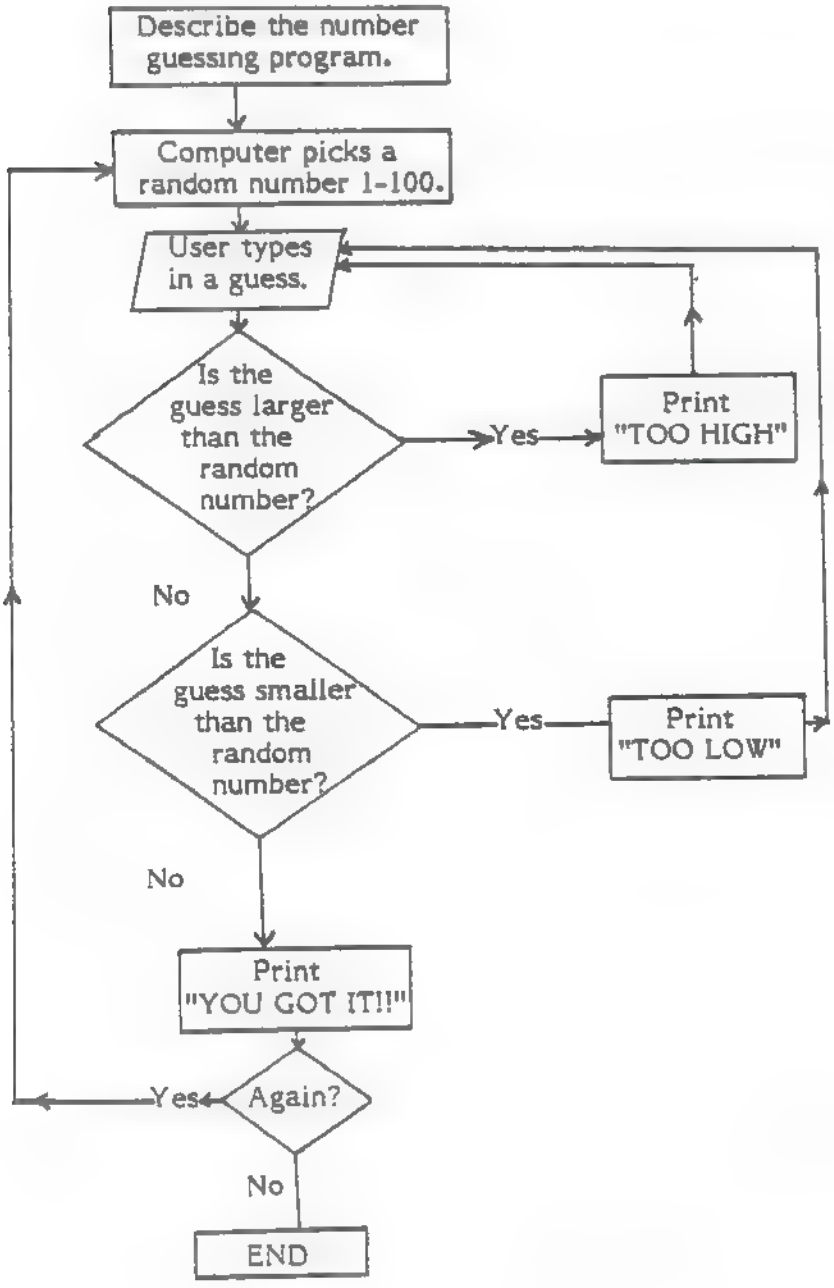
5. Modify the program to ask the user the largest numbers they will get for factors in the multiplication problems. Record the lines you add and change in the space below:

6. SAVE MULTIPLICATION DRILL

## Activities

### Program 4: RND for a number guessing game

1. Write a program for a number guessing game. The computer will randomly select (BUT NOT PRINT!) a number from 1 to 100. The user must input a guess. The computer will print "TOO HIGH" or "TOO LOW" and allow the user another guess or print "YOU GOT IT". Use the flow chart below:




2. Run the program and SAVE NUMBER GUESS

## Activities

### Program 5: GOSUB....RETURN

1. LOAD PROGRAM IV-5 from the Intermediate BASIC disk or type the following:

```
]NEW

110  HOME
120  PRINT "THIS PROGRAM WILL CALCULATE"
130  PRINT "THE CIRCUMFERENCE AND AREA"
140  PRINT "OF CIRCLES GIVEN THE DIAMETER."
150  PRINT
160  INPUT "WHAT IS THE DIAMETER? ";D
170  LET R = D / 2
180  INPUT "UNIT OF MEASURE? ";U$
190  LET P = 3.1416
200  LET C = P * D
210  LET A = P * R * R
220  PRINT "CIRCUMFERENCE = " C " " U$
230  PRINT "AREA = " A " SQUARE " U$
240  PRINT
250  INPUT "AGAIN - Y OR N? ";A$
260  IF  LEFT$ (A$,1) = "Y" THEN 110
270  HOME
280  END
```

2. Run the program. Use the following measurements and record your results below:

| Diameter | Unit | Circumference | Area |
|---|---|---|---|
| 2.4 | Centimeters | | |
| 8.3 | Meters | | |
| 15.9 | Millimeters | | |
| 1.9 | Centimeters | | |

3. Add a subroutine before line 11Ø to round decimals to tenths. Use the subroutine to round the circumference and area to tenths before printing the values. Record the lines added in the space below:

4. Run the program using the same measurements as in the table above. Compare your rounded results to the first results.

## Activities

### Program 6: ON....GOSUB

1. LOAD SAMPLE IV-9 from page IV-9.

2. Add and change lines to make the "THAT IS IT!!!" message one of four random messages.

   Line 32Ø should generate a random number 1 to 4.

   Line 325 should be an ON....GOSUB statement that will branch to a subroutine that will print:

   ...."THAT IS IT!!!", if the random number is 1.

   ...."WAY TO GO!", if the random number is 2.

   ...."YOU GOT IT!", if the random number is 3.

   or ...."WHAT A HUMAN CALCULATOR YOU ARE!", if the random number is 4.

   Four subroutines must be added to print the four different messages. Complete the statements using the line numbers below:

   32Ø LET W = _____

   325 ON W GOSUB ___, ___, ___, ___

   121 PRINT "THAT IS IT!!!"
   122 RETURN

   123 _____
   124 RETURN

   125 _____
   126 RETURN

   127 _____
   128 RETURN

3. Type in the changes and additions and RUN the program.

4. SAVE ( any name you wish! )

## Activities

### Program 7: ON....GOTO

1. Write a program similar to SAMPLE IV-10 on page IV-10. Use an ON....GOTO statement and count the totals that come up in one hundred rolls of a pair of dice. Write your program in the space below:

2. Type in and run your program. Record on the chart below the number of times each total came up:

| 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |

3. Compare your results with someone else. If you were a betting person what total would you bet on? _____

# TEACHER NOTES

## Creating a Demonstration Diskette

The emphasis in the Intermediate units is to modify and enhance sample programs using new commands, then explore and analyze the effects of the changes. To aid this process, it is suggested that the trainer prepare a demonstration diskette to be used with these materials. The diskette should include copies of all sample and activity programs referenced in the units. Save the programs on the diskette using the same program names found in the units.

The diskette would aid both the trainer and participants by reducing the amount of time needed to enter and run programs. The diskette would be especially useful when participants' hands-on computer time is limited.

Suggested solutions to the activities might be placed on a separate diskette.

# UNIT I - Suggested Solutions

Page I-5

Answer these questions:

- Is the HELLO program locked? NO
- How many sectors does the HELLO program occupy? 2 sectors
- How many sectors are remaining for storing other programs and files on your disk? 494 (560 total - 64 for DOS and the directory - 2 for the HELLO program = 494)
- What type of file is the HELLO program? Applesoft

| Type the following. Press RETURN after each line. | Record what happens below. |
|---|---|
| LIST | The HELLO program is listed on the screen. |
| RUN | The HELLO program is executed (run). |
| NEW | The HELLO program is cleared from memory. (A copy of the HELLO program is still stored on disk.) |
| LIST | Nothing is listed since NEW cleared memory of any existing program. |
| RUN | Nothing is run since there is no program in memory. |

# UNIT I - Suggested Solutions

## Page I-11: Activity Program 1

2. (PROGRAM I-1 on the Intermediate BASIC disk.)
RUN

```
MARY                    JOHN                    BILL
16
ANDYCAROLHELLO
```

Using the semicolon in a PRINT statement causes items to be printed right after each other with no spaces between.

3. (SOLUTION I-1A on the Intermediate BASIC disk.)
LIST

```
5  REM  THIS IS A SAMPLE PROGRAM
10  HOME
20  PRINT "MARY","JOHN","BILL"
30  PRINT (1 + 3) ^ 2
40  PRINT "ANDY",
50  PRINT "CAROL",
60  PRINT "HELLO"
70  END
```

RUN

```
MARY                    JOHN                    BILL
16
ANDY                    CAROL                   HELLO
```

Using the comma in a PRINT statement causes items to be printed in one of three tab fields of the screen. The first tab field is columns 1 through 16. The second tab field is columns 17 through 32. The third tab field is columns 33 through 40. ANDY is printed starting in column 1, CAROL in column 17 and HELLO in column 33.

4. (SOLUTION I-1B on the Intermediate BASIC disk.)
RUN

```
MARY                    JOHN                    BILL
16
ANDY                    CAROL                   HELLO
HELLO
HELLO
HELLO
HELLO
HELLO
HELLO
HELLO
HELLO
HELLO

BREAK IN 60
```

# UNIT I - Suggested Solutions

## Page I-12: Activity Program 2

2. (PROGRAM I-2 on the Intermediate BASIC disk.)

RUN

```
9
20
```

3. (SOLUTION I-2A on the Intermediate BASIC disk.)

LIST

```
10  HOME
20 A = 4
30 B = 5
40  LET C = A + B
50  PRINT C
60 PRODUCT = A * B
70  PRINT PRODUCT
80  END
```

RUN

```
9
20
```

Omitting the word LET does not change the output.

4. (SOLUTION I-2B on the Intermediate BASIC disk.)

LIST

```
10  HOME
20 A = 4
30 B = 5
40  LET C = A + B
50  PRINT C
60 PRODUCT = A * B
70  PRINT PRICE
90  END
```

RUN

```
9
20
```

There are no changes in the output.

# UNIT I - Suggested Solutions

## Page I-13: Activity Program 3

2. (PROGRAM I-3 on the Intermediate BASIC disk.)
RUN

```
3
7
7
7
7
```

3. (SOLUTION I-3 on the Intermediate BASIC disk.)
LIST

```
 5  HOME
10 C = 3
15 CA = 15
30 CAD = 6
31 CAC = 7
32  PRINT C
33  PRINT CA
34  PRINT CAB
35  PRINT CAD
36 CAB = 4
37  PRINT CAC
50  END
```

RUN

```
3
7
7
7
4
```

The last value printed is 4 rather than 7.
Line 36 changes the value of CA from 7 to 4.

## UNIT I - Suggested Solutions

### Page I-14: Activity Program 4

2. (PROGRAM I-4 on the Intermediate BASIC disk.)
RUN (Input from the keyboard is underlined.)

```
WHAT IS YOUR NAME ?NORMAN
HI, NORMAN.
THIS IS YOUR LUCKY DAY, NORMAN.
```

3. (SOLUTION I-4A on the Intermediate BASIC disk.)
LIST

```
10  HOME
20  INPUT "WHAT IS YOUR NAME";B$
40  PRINT "HI, ";B$"."
50  LET C$ = "THIS IS YOUR LUCKY DAY, "
60  PRINT C$;B$"."
70  END
```

RUN (Input from the keyboard is underlined.)

```
WHAT IS YOUR NAMENORMAN
HI, NORMAN.
THIS IS YOUR LUCKY DAY, NORMAN.
```

There is no question mark following WHAT IS YOUR NAME.

(SOLUTION I-4B on the Intermediate BASIC disk.)
To make the question mark appear, change line 20 as follows:

```
20 INPUT "WHAT IS YOUR NAME?  "; B$
```

RUN (Input from the keyboard is underlined.)

```
WHAT IS YOUR NAME? NORMAN
HI, NORMAN.
THIS IS YOUR LUCKY DAY, NORMAN.
```

# UNIT I - Suggested Solutions

## Page I-15: Acitivity Program 5

2. (PROGRAM I-5 on the Intermediate BASIC disk.)
RUN (Input from the keyboard is underlined.)

```
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  1
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  2
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  3
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  4
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  0
TOTAL = 10
```

3. (SOLUTION I-5A on the Intermediate BASIC disk.)
LIST

```
10  HOME
20  LET T = 0
30  PRINT "TYPE 0 TO HAVE THE TOTAL PRINTED."
40  INPUT "TYPE A NUMBER:  ";N
50  IF N = 0 GOTO 80
60  LET T = T + N
70  GOTO 30
80  PRINT "TOTAL = "T
90  END
```

RUN (Input from the keyboard is underlined.)

```
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  1
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  2
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  3
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  4
TYPE 0 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  0
TOTAL = 10
```

There are no differences in the run of the program.

4. (SOLUTION I-5B on the Intermediate BASIC disk.)
Change the following lines to make -1 the number to type to have the total printed:

```
30 PRINT "TYPE -1 TO HAVE THE TOTAL PRINTED."
50 IF N = -1 THEN 80
```

LIST

```
10  HOME
20  LET T = 0
30  PRINT "TYPE -1 TO HAVE THE TOTAL PRINTED."
40  INPUT "TYPE A NUMBER:  ";N
50  IF N =  - 1 THEN 80
60  LET T = T + N
70  GOTO 30
80  PRINT "TOTAL = "T
90  END
```

Page I-15: Activity Program 5 (continued)

RUN (Input from the keyboard is underlined.)

```
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  1
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  2
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  3
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  4
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  0
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  5
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  -1
TOTAL = 15
```

5. (SOLUTION I-5C on the Intermediate BASIC disk.)
Add lines similar to the following to count the numbers and print the average:

```
25 LET C = 0
65 LET C = C + 1
85 PRINT "COUNT  =  " C
87 PRINT "AVERAGE  =  " T/C
```

LIST

```
10  HOME
20  LET T = 0
25  LET C = 0
30  PRINT "TYPE -1 TO HAVE THE TOTAL PRINTED."
40  INPUT "TYPE A NUMBER:  ";N
50  IF N =  - 1 THEN 80
60  LET T = T + N
65  LET C = C + 1
70  GOTO 30
80  PRINT "TOTAL = "T
85  PRINT "COUNT = "C
87  PRINT "AVERAGE = "T / C
90  END
```

RUN (Input from the keyboard is underlined.)

```
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  1
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  2
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  3
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  4
TYPE -1 TO HAVE THE TOTAL PRINTED.
TYPE A NUMBER:  -1
TOTAL = 10
COUNT = 4
AVERAGE = 2.5
```

## UNIT I - Suggested Solutions

### Page I-16: Activity Program 6

2. (PROGRAM I-6 on the Intermediate BASIC disk.)

RUN

```
 5                    6
ANN                   ANDY
```

3. (SOLUTION I-6A on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  READ A, B
30  PRINT A, B
40  READ C$, D$
50  PRINT C$, D$
60  DATA  5, 6, "ANN", "ANDY"
70  END
```

RUN

```
 5                    6
ANN                   ANDY
```

Using quotation marks with string values in DATA statements did not change the output.

4. (SOLUTION I-6B on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  READ A, B
30  PRINT A, B
40  READ C$, D$
50  PRINT C$, D$
60  DATA  5, 6, ST. PAUL, MN, HUNTLEY, MN
70  END
```

RUN

```
 5                    6
ST. PAUL              MN
```

The READ command takes an element from the DATA. The Apple uses commas to separate items in the DATA statements. In line 40, ST. PAUL is read into C$ and MN is read into D$.

## Page I-16: Activity Program 6 (continued)

(SOLUTION I-6C on the Intermediate BASIC disk.)
LIST

```
10  HOME
20  READ A, B
30  PRINT A, B
40  READ C$, D$
50  PRINT C$, D$
60  DATA  5, 6, "ST. PAUL, MN", "HUNTLEY, MN"
70  END
```

RUN

```
5                    6
ST. PAUL, MN         HUNTLEY, MN
```

When a quotation mark is the first character of a string DATA element the Apple reads all characters that follow until the closing quotation mark. Commas between the quotation marks are included as part of the string. In line 40 the Apple reads ST. PAUL, MN into C$ and HUNTLEY, MN into D$.

## Page I-17: Activity Program 7

2. (PROGRAM I-7 on the Intermediate BASIC disk.)
RUN

```
NUMBER = 0
NUMBER = 1
NUMBER = 2
NUMBER = 3
NUMBER = 4
NUMBER = 5
NUMBER = 6
NUMBER = 7
NUMBER = 8
NUMBER = 9
NUMBER = 10
NUMBER = 11
NUMBER = 12
```

Page I-17: Activity Program 7 (continued)

3. (SOLUTION I-7A on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  FOR NUMBER = 0 TO 12 STEP 2
30  PRINT "NUMBER = "NUMBER
40  NEXT NUMBER
50  END
```

RUN

```
NUMBER = 0
NUMBER = 2
NUMBER = 4
NUMBER = 6
NUMBER = 8
NUMBER = 10
NUMBER = 12
```

The value of NUMBER is increased by 2 each time through the loop. Only even numbers 0 through 12 are printed.

(SOLUTION I-7B on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  FOR NUMBER = 12 TO 0 STEP  - 1
30  PRINT "NUMBER = "NUMBER
40  NEXT NUMBER
50  END
```

RUN

```
NUMBER = 12
NUMBER = 11
NUMBER = 10
NUMBER = 9
NUMBER = 8
NUMBER = 7
NUMBER = 6
NUMBER = 5
NUMBER = 4
NUMBER = 3
NUMBER = 2
NUMBER = 1
NUMBER = 0
```

Each time through the loop, 1 is subtracted from the value of NUMBER. The Apple counts down from 12 to 0.

UNIT I - Suggested Solutions

## Page I-18: Activity Program 8

2. (PROGRAM I-8 on the Intermediate BASIC disk.)
RUN

```
THIS IS AS SLOW AS I CAN GO.
SPEED 150 IS NICE FOR READING.
THIS IS NORMAL SPEED.
```

Printed at SPEED = 0

Printed at SPEED = 150

Printed at SPEED = 255

4. (SOLUTION I-8 on the Intermediate BASIC disk.)
LIST

```
2  HOME
5  SPEED= 0
7  INVERSE
10  PRINT "THIS IS AS SLOW AS I CAN GO."
15  NORMAL
20  SPEED= 150
30  PRINT "SPEED 150 IS NICE FOR READING."
40  SPEED= 255
45  FLASH
50  PRINT "THIS IS NORMAL SPEED."
55  NORMAL
60  END
```

RUN

```
THIS IS AS SLOW AS I CAN GO.
SPEED 150 IS NICE FOR READING.
THIS IS NORMAL SPEED.
```

This line appears as black characters on a white background (INVERSE).

This line is printed as white on a black background.

This line is flashing on the screen.

## UNIT II - Suggested Solutions

Page II-1

2. (SAMPLE II-1C on the Intermediate BASIC disk.)

To print cubes of the numbers 11 through 20, change these lines:

```
20 PRINT "NUMBER",  "  CUBE"
50 FOR N = 11 TO 20
60 PRINT N, N*N*N
```

LIST

```
10  HOME
20  PRINT "NUMBER","  CUBE "
30  PRINT "------","------"
40  PRINT
50  FOR N = 11 TO 20
60  PRINT N, N * N * N
90  NEXT N
100   PRINT "I AM FINISHED LOOPING!"
110   END
```

RUN

```
NUMBER              CUBE
------              ------

11                  1331
12                  1728
13                  2197
14                  2744
15                  3375
16                  4096
17                  4913
18                  5832
19                  6859
20                  8000
I AM FINISHED LOOPING!
```

3. (SAMPLE II-1D on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  PRINT "NUMBER","  CUBE "
30  PRINT "------","------"
40  PRINT
50  FOR N = 2 TO 20 STEP 2
60  PRINT N, N * N * N
90  NEXT N
100   PRINT "I AM FINISHED LOOPING!"
110   END
```

## UNIT II - Suggested Solutions

Page II-1 (continued)

RUN

```
NUMBER          CUBE
------          ------

2               8
4               64
6               216
8               512
10              1000
12              1728
14              2744
16              4096
18              5832
20              8000
I AM FINISHED LOOPING!
```

4. (SAMPLE II-1E on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  PRINT "NUMBER","  CUBE "
30  PRINT "------","------"
40  PRINT
50  FOR N = 10 TO 0 STEP  - 1
60  PRINT N,N * N * N
90  NEXT N
100  PRINT "I AM FINISHED LOOPING!"
110  END
```

RUN

```
NUMBER          CUBE
------          ------

10              1000
9               729
8               512
7               343
6               216
5               125
4               64
3               27
2               8
1               1
0               0
I AM FINISHED LOOPING!
```

## UNIT II - Suggested Solutions

### Page II-3

Make these changes:

```
 90 NEXT B
110 NEXT R
```

What happens? The Apple "beeps" and prints the message "? NEXT WITHOUT FOR ERROR IN 110."

Why? The B loop and the R loop cross. The Apple completes the B loop and looks outside the body of the B loop for a FOR statement to go with the NEXT R in line 110.

(SAMPLE II-3A on the Intermediate BASIC disk.)
To make the Apple repeat drawing the three blocks two times, add the following lines:

```
 35 FOR T = 1 TO 2
115 NEXT T
```

LIST

```
10  HOME
11  SPEED= 175
20  PRINT "THIS WILL PRINT 3 BLOCKS"
30  PRINT "WITH 2 ROWS OF 4 STARS."
35  FOR T = 1 TO 2
40  PRINT
50  FOR B = 1 TO 3
60  PRINT "BLOCK #"B
70  FOR R = 1 TO 2
80  PRINT "****"
90  NEXT R
100  PRINT
110  NEXT B
115  NEXT T
119  SPEED= 255
120  END
```

RUN

```
THIS WILL PRINT 3 BLOCKS
WITH 2 ROWS OF 4 STARS.

BLOCK #1
****
****

BLOCK #2
****
****

BLOCK #3
****
****


BLOCK #1
****
****

BLOCK #2
****
****

BLOCK #3
****
****
```

### Page II-4

(SAMPLE II-4 on the Intermediate BASIC disk.)
Change the following line to make the program loop 5 times:

```
50 FOR S = 1 TO 5
```

What message does the Apple give you? ? OUT OF DATA ERROR IN 60

Page II-6

(SAMPLE II-5A on the Intermediate BASIC disk.)

LIST

```
 5  SPEED= 150
 10  HOME
 20  PRINT "PROGRAM TO AVERAGE"
 30  PRINT "PTS. ON 3 TESTS."
 40  FOR S = 1 TO 4
 50  READ N$, A, B, C
 60  LET T = A + B + C
 70  PRINT "======================================="
 80  PRINT "NAME:   "N$
 90  PRINT "TOTAL:   "T,"AVERAGE:   "T / 3
 100  NEXT S
 110  PRINT
 120  PRINT "TEST SCORES"
 130  FOR X = 1 TO 4
 140  READ N$, A, B, C
 150  PRINT "NAME:   "N$
 160  PRINT "A="A, "B="B, "C="C
 170  PRINT
 180  NEXT X
 190  SPEED= 255
 200  END
 500  DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
 510  DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

Run the program.

What message do you get? ? OUT OF DATA ERROR IN 140

(POINTS on the Intermediate BASIC disk.)

LIST

```
 5  SPEED= 150
 10  HOME
 20  PRINT "PROGRAM TO AVERAGE"
 30  PRINT "PTS. ON 3 TESTS."
 40  FOR S = 1 TO 4
 50  READ N$, A, B, C
 60  LET T = A + B + C
 70  PRINT "======================================="
 80  PRINT "NAME:   "N$
 90  PRINT "TOTAL:   "T,"AVERAGE:   "T / 3
 100  NEXT S
 105  RESTORE
 110  PRINT
 120  PRINT "TEST SCORES"
 130  FOR X = 1 TO 4
 140  READ N$, A, B, C
 150  PRINT "NAME:   "N$
 160  PRINT "A="A, "B="B, "C="C
 170  PRINT
 180  NEXT X
 190  SPEED= 255
 200  END
 500  DATA  JO, 9, 8, 10, MARIEN, 9, 10, 10
 510  DATA  ERNIE, 5, 7, 9, ALICE, 10, 3, 5
```

## UNIT II - Suggested Solutions

Page II-6 (continued)

RUN

```
PROGRAM TO AVERAGE
PTS. ON 3 TESTS.
=====================================
NAME:  JO
TOTAL:  27      AVERAGE:  9
=====================================
NAME:  MARIEN
TOTAL:  29      AVERAGE:  9.66666667
=====================================
NAME:  ERNIE
TOTAL:  21      AVERAGE:  7
=====================================
NAME:  ALICE
TOTAL:  18      AVERAGE:  6

TEST SCORES
NAME:  JO
A=9             B=8             C=10

NAME:  MARIEN
A=9             B=10            C=10

NAME:  ERNIE
A=5             B=7             C=9

NAME:  ALICE
A=10            B=3             C=5
```

Page II-7

What is the ending balance? $695.17

Is the ending balance $350 more? Yes. It is $1045.17.

## UNIT II - Suggested Solutions

### Page II-9: Activity Program 1

1. (SOLUTION II-1A on the Intermediate BASIC disk.)

LIST

```
 10  HOME
 20  FOR A = 1 TO 3
 30  PRINT
 40  PRINT "XXXXX"
 50  FOR B = 1 TO 4
 60  PRINT "X   X"
 70  NEXT B
 80  PRINT "XXXXX"
 90  NEXT A
100  END
```

2. (SOLUTION II-1B on the Intermediate BASIC disk.)

LIST

```
 10  HOME
 30  PRINT
 40  PRINT "XXXXX","XXXXX","XXXXX"
 50  FOR B = 1 TO 12
 60  PRINT "X   X",
 70  NEXT B
 80  PRINT "XXXXX","XXXXX","XXXXX"
100  END
```

OR .... (SOLUTION II-1C on the Intermediate BASIC disk.)

LIST

```
 10  HOME
 20  FOR A = 1 TO 3
 40  PRINT "XXXXX",
 45  NEXT A
 50  FOR B = 1 TO 12
 60  PRINT "X   X",
 70  NEXT B
 75  FOR C = 1 TO 3
 80  PRINT "XXXXX",
 90  NEXT C
100  END
```

## UNIT II - Suggested Solutions

### Page II-10: Activity Program 2

2. (PROGRAM II-2 on the Intermediate BASIC disk.)

RUN

```
THE PRODUCT OF THE NUMBERS
FROM 1 TO 5 IS 120.

THE PRODUCT OF THE NUMBERS
FROM 1 TO 8 IS 40320.

THE PRODUCT OF THE NUMBERS
FROM 1 TO 12 IS 479001600.
```

3. Change the following lines to produce the sum of the numbers rather than the product:

```
40 LET P = 0
60 LET P = P + N
80 PRINT "THE SUM OF THE NUMBERS"
```

4. Change the following lines to produce the sums of the numbers 1 to 10, 1 to 100, 1 to 50 and 1 to 500:

```
20 FOR T = 1 TO 4
120 DATA 10, 100, 50 500
```

5. (SOLUTION II-2A on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  FOR T = 1 TO 4
30  READ R
40  LET P = 0
50  FOR N = 1 TO R
60  LET P = P + N
70  NEXT N
80  PRINT "THE SUM OF THE NUMBERS"
90  PRINT "FROM 1 TO "R" IS "P"."
100  PRINT
110  NEXT T
120  DATA  10,100,50,500
130  END
```

RUN

```
THE SUM OF THE NUMBERS
FROM 1 TO 10 IS 55.

THE SUM OF THE NUMBERS
FROM 1 TO 100 IS 5050.

THE SUM OF THE NUMBERS
FROM 1 TO 50 IS 1275.

THE SUM OF THE NUMBERS
FROM 1 TO 500 IS 125250.
```

| Numbers | Sum |
|---|---|
| 1 to 10 | 55 |
| 1 to 100 | 5050 |
| 1 to 50 | 1275 |
| 1 to 500 | 125250 |

Page II-10: Activity Program 2 (continued)

6. (SOLUTION II-2B on the Intermediate BASIC disk.)

LIST

```
10  HOME
20  FOR T = 1 TO 2
30  READ R
40  LET P = 0
50  FOR N = 1 TO R
60  LET P = P + N
70  NEXT N
80  PRINT "THE SUM OF THE NUMBERS"
90  PRINT "FROM 1 TO "R" IS "P"."
100  PRINT
110  NEXT T
120  DATA  1000, 10000
130  END
```

***NOTE: Be patient! It takes the Apple about 1 minute to print the sums.

RUN

```
THE SUM OF THE NUMBERS
FROM 1 TO 1000 IS 500500.

THE SUM OF THE NUMBERS
FROM 1 TO 10000 IS 50005000.
```

| Numbers | Sum | |
|---|---|---|
| | Prediction | Actual |
| 1 to 1000 | | 500500 |
| 1 to 10000 | | 50005000 |

## UNIT II - Suggested Solutions

### Page II-11: Activity Program 3

2. (SOLUTION II-3 on the Intermediate BASIC disk.)

LIST

(Lines changed are starred.)

```
  5  SPEED= 150
  10  HOME
  20  PRINT "PROGRAM TO AVERAGE"
  30  PRINT "PTS. ON 3 TESTS."
*40  FOR S = 1 TO 5
  50  READ N$, A, B, C
  60  LET T = A + B + C
  70  PRINT "=================================="
  80  PRINT "NAME:  "N$
  90  PRINT "TOTAL:  "T,"AVERAGE:  "T / 3
  100  NEXT S
  105  RESTORE
  110  PRINT
  120  PRINT "TEST SCORES"
*130  FOR X = 1 TO 5
  140  READ N$, A, B, C
  150  PRINT "NAME:  "N$
  160  PRINT "A="A,"B="B,"C="C
  170  PRINT
  180  NEXT X
  190  SPEED= 255
  200  END
*500  DATA  HILDA, 7, 12, 15
*510  DATA  BART, 13, 7, 14
*520  DATA  JENNY, 12, 12, 15
*530  DATA  ARTHUR, 15, 15, 15
*540  DATA  MARTHA, 14, 10, 15
```

RUN

```
PROGRAM TO AVERAGE
PTS. ON 3 TESTS.
==================================
NAME:  HILDA
TOTAL:  34       AVERAGE:  11.3333333
==================================
NAME:  BART
TOTAL:  34       AVERAGE:  11.3333333
==================================
NAME:  JENNY
TOTAL:  39       AVERAGE:  13
==================================
NAME:  ARTHUR
TOTAL:  45       AVERAGE:  15
==================================
NAME:  MARTHA
TOTAL:  39       AVERAGE:  13

TEST SCORES
NAME:  HILDA
A=7              B=12             C=15

NAME:  BART
A=13             B=7              C=14

NAME:  JENNY
A=12             B=12             C=15

NAME:  ARTHUR
A=15             B=15             C=15

NAME:  MARTHA
A=14             B=10             C=15
```

## UNIT II - Suggested Solutions

### Page II-11: Activity Program 4

1. (CHECKBOOK on the Intermediate BASIC disk.)

LIST

(Lines added are starred.)

```
  10  HOME
  20  PRINT "THIS IS A CHECKBOOK BALANCING PROGRAM"
  30  PRINT "<> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <>"
  40  REM  *** BALANCE ***
  50  READ B
  60  PRINT "BEGINNING BALANCE = $"B
  70  REM  *** DEPOSITS ***
  80  READ D
  90  IF D = 0 THEN 130
 100   LET B = B + D
 110   GOTO 80
 120   REM  *** CHECKS ***
 130   READ C
 140   IF C = 0 THEN 170
 150   LET B = B - C
 160   GOTO 130
 170   PRINT "ENDING BALANCE = $"B
*180   RESTORE
*190   READ B
*200   READ D
*210   IF D = 0 THEN 240
*220   PRINT "DEPOSIT = $"D
*230   GOTO 200
*240   READ C
*250   IF C = 0 THEN 280
*260   PRINT "CHECK = $"C
*270   GOTO 240
*280   END
 500   DATA  593.83
 510   DATA  278.04,12,0
 520   DATA  9,41.65,7,50,12.02,57.77,11.26,0
```

RUN

```
THIS IS A CHECKBOOK BALANCING PROGRAM
<> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <> <>
BEGINNING BALANCE = $593.83
ENDING BALANCE = $695.17
DEPOSIT = $278.04
DEPOSIT = $12
CHECK = $9
CHECK = $41.65
CHECK = $7
CHECK = $50
CHECK = $12.02
CHECK = $57.77
CHECK = $11.26
```

## Page II-12: Activity Program 5

2. (PROGRAM II-5 on the Intermediate BASIC disk.)

RUN

```
*** ALLEY CATS BOWLING TEAM ***
====================================
PLAYER:  MARY
SCORES=150       148              219
TOTAL= 517       AVERAGE= 172.333333
SCORES=155       145              165
TOTAL= 465       AVERAGE= 155
GRAND TOTAL = 982
NUMBER OF GAMES = 6
OVERALL AVERAGE = 163.666667
====================================
PLAYER:  JO
SCORES=170       130              100
TOTAL= 400       AVERAGE= 133.333333
SCORES=140       160              160
TOTAL= 460       AVERAGE= 153.333333
GRAND TOTAL = 860
NUMBER OF GAMES = 6
OVERALL AVERAGE = 143.333333
====================================
```

3. (SOLUTION II-5A on the Intermediate BASIC disk.)

LIST

(Lines added are starred.)

```
 10  HOME
 20  LET L$ = "===================================="
 30  PRINT "*** ALLEY CATS BOWLING TEAM ***"
 40  PRINT L$
 50  FOR P = 1 TO 2
 60  LET GT = 0: LET NG = 0
 70  READ N$
 80  PRINT "PLAYER:  "N$
 90  READ G1,G2,G3
 100  LET T = G1 + G2 + G3
 110  LET GT = GT + T
 120  LET NG = NG + 3
 130  PRINT "SCORES="G1,G2,G3
 140  PRINT "TOTAL= "T,"AVERAGE= "T / 3
 150  READ F
 160  IF F =  - 1 THEN 90
 170  PRINT "GRAND TOTAL = "GT
 180  PRINT "NUMBER OF GAMES = "NG
 190  PRINT "OVERALL AVERAGE = "GT / NG
 200  PRINT L$
 210  NEXT P
 300  END
 1000  DATA  MARY
 1001  DATA  150,148,219
 1002  DATA  -1
 1003  DATA  155,145,165
*1004  DATA  -1
*1005  DATA  188,200,175
*1006  DATA  -2
 2000  DATA  JO
 2001  DATA  170,130,100
 2002  DATA  -1
 2003  DATA  140,160,160
*2004  DATA  -1
*2005  DATA  155,130,177
*2006  DATA  -2
```

RUN

```
*** ALLEY CATS BOWLING TEAM ***
====================================
PLAYER:  MARY
SCORES=150       148              219
TOTAL= 517       AVERAGE= 172.333333
SCORES=155       145              165
TOTAL= 465       AVERAGE= 155
SCORES=188       200              175
TOTAL= 563       AVERAGE= 187.666667
GRAND TOTAL = 1545
NUMBER OF GAMES = 9
OVERALL AVERAGE = 171.666667
====================================
PLAYER:  JO
SCORES=170       130              100
TOTAL= 400       AVERAGE= 133.333333
SCORES=140       160              160
TOTAL= 460       AVERAGE= 153.333333
SCORES=155       130              177
TOTAL= 462       AVERAGE= 154
GRAND TOTAL = 1322
NUMBER OF GAMES = 9
OVERALL AVERAGE = 146.888889
====================================
```

## UNIT II - Suggested Solutions

### Page II-12: Activity Program 5 (continued)

4. (SOLUTION II-5B on the Intermediate BASIC disk.)

LIST

(Lines added and changed are starred.)

```
  10  HOME
  20  LET L$ = "========================================"
  30  PRINT "**** ALLEY CATS BOWLING TEAM ****"
  40  PRINT L$
* 50  FOR P = 1 TO 4
  60  LET GT = 0: LET NG = 0
  70  READ N$
  80  PRINT "PLAYER:  "N$
  90  READ G1,G2,G3
  100  LET T = G1 + G2 + G3
  110  LET GT = GT + T
  120  LET NG = NG + 3
  130  PRINT "SCORES="G1,G2,G3
  140  PRINT "TOTAL= "T,"AVERAGE= "T / 3
  150  READ F
  160  IF F =  - 1 THEN 90
  170  PRINT "GRAND TOTAL = "GT
  180  PRINT "NUMBER OF GAMES = "NG
  190  PRINT "OVERALL AVERAGE = "GT / NG
  200  PRINT L$
  210  NEXT P
  300  END
  1000  DATA  MARY
  1001  DATA  150,148,219
  1002  DATA  -1
  1003  DATA  155,145,165
  1004  DATA  -1
  1005  DATA  188,200,175
  1006  DATA  -2
  2000  DATA  JO
  2001  DATA  170,130,100
  2002  DATA  -1
  2003  DATA  140,160,160
  2004  DATA  -1
  2005  DATA  155,138,177
  2006  DATA  -2
* 3000  DATA  JAN
* 3001  DATA  180,165,140
* 3002  DATA  -1
* 3003  DATA  133,144,122
* 3004  DATA  -1
* 3005  DATA  150,140,150
* 3006  DATA  -2
* 4000  DATA  DOUG
* 4001  DATA  179,164,139
* 4002  DATA  -1
* 4003  DATA  134,145,150
* 4004  DATA  -1
* 4005  DATA  201,180,300
* 4006  DATA  -2
```

RUN

```
**** ALLEY CATS BOWLING TEAM ****
========================================
PLAYER:  MARY
SCORES=150      148             219
TOTAL= 517      AVERAGE= 172.333333
SCORES=155      145             165
TOTAL= 465      AVERAGE= 155
SCORES=188      200             175
TOTAL= 563      AVERAGE= 187.666667
GRAND TOTAL = 1545
NUMBER OF GAMES = 9
OVERALL AVERAGE = 171.666667
========================================
PLAYER:  JO
SCORES=170      130             100
TOTAL= 400      AVERAGE= 133.333333
SCORES=140      160             160
TOTAL= 460      AVERAGE= 153.333333
SCORES=155      138             177
TOTAL= 462      AVERAGE= 154
GRAND TOTAL = 1322
NUMBER OF GAMES = 9
OVERALL AVERAGE = 146.888889
========================================
PLAYER:  JAN
SCORES=180      165             140
TOTAL= 485      AVERAGE= 161.666667
SCORES=133      144             122
TOTAL= 399      AVERAGE= 133
SCORES=150      140             150
TOTAL= 440      AVERAGE= 146.666667
GRAND TOTAL = 1324
NUMBER OF GAMES = 9
OVERALL AVERAGE = 147.111111
========================================
PLAYER:  DOUG
SCORES=179      164             139
TOTAL= 482      AVERAGE= 160.666667
SCORES=134      145             150
TOTAL= 429      AVERAGE= 143
SCORES=201      180             300
TOTAL= 681      AVERAGE= 227
GRAND TOTAL = 1592
NUMBER OF GAMES = 9
OVERALL AVERAGE = 176.888889
========================================
```

# UNIT III - Suggested Solutions

## Page III-6

Change these lines in SAMPLE III-5:

```
30 GET T
50 PRINT "YOU PRESSED:    " T
```

Run the program and press a letter key.

What happens? The Apple "beeps" and prints the message ? SYNTAX ERROR.

## Page III-7

When trying to input a string longer than 255 characters, the following happens:

The Apple "beeps" when the 249th character is typed.

The Apple continues to beep each time an additional character is typed.

When the 256th character is typed the Apple prints a "\" and forgets everything you have just typed.

When RETURN is pressed, to end the input, only the characters typed after the "\" are stored as the string value.

Change the following lines SAMPLE III-6:

```
20 PRINT "TYPE A NUMBER."
30 INPUT N
35 LET S$ = STR$(N)
```

RUN (Input from the keyboard is underlined).

```
TYPE A NUMBER.
?+12345

YOU TYPED:
 12345

THERE ARE 5 CHARACTERS AND SPACES.

AGAIN?  Y OR N


FAREWELL!
```

Try the following numbers and record the number of characters in each:

| | | | | | |
|---|---|---|---|---|---|
| 37 | 2 characters | .2815 | 5 characters | -12345 | 6 characters |
| 37.8 | 4 characters | 0.2815 | 5 characters | +12345 | 5 characters |

Page III-8

Change this line to have the first 3 characters typed:

```
60 PRINT "THE FIRST THREE LETTERS ARE  " LEFT$ (W$,3) "."
```

RUN (Input from the keyboard is underlined.)

```
TYPE A WORD:  ?MICROCOMPUTER

YOU TYPED:  MICROCOMPUTER
IT HAS 13 CHARACTERS.
THE FIRST THREE LETTERS ARE MIC.
AGAIN? YES


TYPE A WORD:  ?IT

YOU TYPED:  IT
IT HAS 2 CHARACTERS.
THE FIRST THREE LETTERS ARE IT.
AGAIN? NO
SEE YA!
```

Change this line to have the last two letters typed:

```
60 PRINT "THE LAST TWO LETTERS ARE  " RIGHT $ (W$,2) "."
```

RUN (Input from the keyboard is underlined.)

```
TYPE A WORD:  ?MICROCOMPUTER

YOU TYPED:  MICROCOMPUTER
IT HAS 13 CHARACTERS.
THE LAST TWO LETTERS ARE ER.
AGAIN? YES


TYPE A WORD:  ?A

YOU TYPED:  A
IT HAS 1 CHARACTERS.
THE LAST TWO LETTERS ARE A.
AGAIN? NO
SEE YA!
```

Page III-9

(SAMPLE III-8B on the Intermediate BASIC disk.)
RUN (Input from the keyboard is underlined.)

```
TYPE A NUMBER WITH A DECIMAL POINT.
?4.7890


THE DECIMAL POINT IS CHARACTER 2.
THERE ARE 1 CHARACTERS BEFORE
AND 4 AFTER THE DECIMAL POINT.
AGAIN?  Y
TYPE A NUMBER WITH A DECIMAL POINT.
?.9


THE DECIMAL POINT IS CHARACTER 1.
THERE ARE 0 CHARACTERS BEFORE
AND 1 AFTER THE DECIMAL POINT.
AGAIN?  Y
TYPE A NUMBER WITH A DECIMAL POINT.
?0.9


THE DECIMAL POINT IS CHARACTER 2.
THERE ARE 1 CHARACTERS BEFORE
AND 1 AFTER THE DECIMAL POINT.
AGAIN?  Y
TYPE A NUMBER WITH A DECIMAL POINT.
?-3.768


THE DECIMAL POINT IS CHARACTER 3.
THERE ARE 2 CHARACTERS BEFORE
AND 3 AFTER THE DECIMAL POINT.
AGAIN?  Y
TYPE A NUMBER WITH A DECIMAL POINT.
?+5.5


THE DECIMAL POINT IS CHARACTER 3.
THERE ARE 2 CHARACTERS BEFORE
AND 1 AFTER THE DECIMAL POINT.
AGAIN?  Y
TYPE A NUMBER WITH A DECIMAL POINT.
?298


THERE IS NO DECIMAL POINT IN 298.
AGAIN?  N
```

# UNIT III - Suggested Solutions

## Page III-10: Activity Program 1

(SOLUTION III-1 on the Intermediate BASIC disk.)
LIST

```
10  HOME
20  PRINT  TAB( 14)"MARCIA HORN"
30  PRINT  TAB( 16)"ROOKIE"
40  PRINT  TAB( 11)"633-9110 EXT. 195"
50  END
```

## Page III-10: Activity Program 2

2. (PROGRAM III-2 on the Intermediate BASIC disk.)
RUN (Input from the keyboard is underlined.)

Screen 1:

```
PLEASE TYPE YOUR FIRST NAME.
?BLAISE
PLEASE TYPE YOUR LAST NAME.
?PASCAL
```

Screen 2:

```
PASCAL
BLAISE
```

3. (SOLUTION III-2 on the Intermediate BASIC disk.)
LIST

```
10  HOME
20  PRINT "PLEASE TYPE YOUR FIRST NAME."
30  INPUT F$
40  PRINT "PLEASE TYPE YOUR LAST NAME."
50  INPUT L$
60  HOME
65  VTAB 1
66  HTAB 10
70  PRINT L$
75  VTAB 22
76  HTAB 10
90  PRINT F$
90  END
```

RUN (Input at the keyboard is underlined)

Screen 1:

```
PLEASE TYPE YOUR FIRST NAME.
?BLAISE
PLEASE TYPE YOUR LAST NAME.
?PASCAL
```

Screen 2:

```
PASCAL


BLAISE
```

## UNIT III - Suggested Solutions

### Page III-11: Activity Program 3

(SOLUTION III-3 on the Intermediate BASIC disk.)

| Make the Apple do the following. | BASIC statement to do it. |
|---|---|
| Clear the screen. | `10 HOME` |
| Print "GET READY". | `20 PRINT "GET READY."` |
| Pause. | `30 FOR P = 1 TO 1500: NEXT P` |
| Clear the screen. | `40 HOME` |
| Print "GET SET". | `50 PRINT "GET SET."` |
| Pause. | `60 FOR P = 1 TO 1500: NEXT P` |
| Clear the screen. | `70 HOME` |
| Flash "GO!" on the screen. | `80 FLASH : PRINT "GO!": NORMAL` |
| Pause. | `90 FOR P = 1 TO 1500: NEXT P` |
| Clear the screen. | `100 HOME` |
| Print a good-bye message. | `110 PRINT "GOOD-BYE!"`<br>`120 END` |

RUN

Screen 1:

```
GET READY.
```

Screen 2:

```
GET SET.
```

Screen 3:

```
GO!
```

Screen 4:

```
GOOD-BYE!
```

## UNIT III - Suggested Solutions

### Page III-11: Activity Program 4

1. (SOLUTION III-4 on the Intermediate BASIC disk.)
Change these lines:

```
30 VTAB 22: PRINT "PRESS ANY KEY TO GO ON ..."; :GET K$
60 VTAB 22: PRINT "PRESS ANY KEY TO GO ON ..."; :GET K$
90 VTAB 22: PRINT "PRESS ANY KEY TO GO ON ..."; :GET K$
```

LIST

```
10  HOME
20  PRINT "GET READY."
30  VTAB 22: PRINT "PRESS ANY KEY TO GO ON . . .";: GET K$
40  HOME
50  PRINT "GET SET."
60  VTAB 22: PRINT "PRESS ANY KEY TO GO ON . . .";: GET K$
70  HOME
80  FLASH : PRINT "GO!": NORMAL
90  VTAB 22: PRINT "PRESS ANY KEY TO GO ON . . .";: GET K$
100  HOME
110  PRINT "GOOD-BYE!"
120  END
```

RUN

Screen 1:

```
GET READY.



PRESS ANY KEY TO GO ON . . .
```

Screen 2:

```
GET SET.



PRESS ANY KEY TO GO ON . . .
```

Screen 3:

```
GO!



PRESS ANY KEY TO GO ON . . .
```

Screen 4:

```
GOOD-BYE!
```

## UNIT III - Suggested Solutions

### Page III-12: Activity Program 5

(SOLUTION III-5 on the Intermediate BASIC disk.)

LIST

```
 10  HOME
 20  INPUT "WHAT IS YOUR NAME? ";N$
 30  PRINT
 40  PRINT "HI, "N$"."
 50  PRINT
 60  PRINT "PRESS ANY KEY TO CONTINUE";: GET K$
 70  HOME
 80  PRINT "HOPE YOU ARE HAVING FUN WITH APPLESOFT"
 90  PRINT "INTERMEDIATE BASIC."
 100  PRINT
 110  PRINT "DO YOU UNDERSTAND THE GET STATEMENT?"
 120  PRINT
 130  PRINT "PLEASE TYPE Y FOR YES AND N FOR NO: ";
 140  GET A$
 150  PRINT
 160  PRINT
 170  IF A$ = "Y" THEN 200
 180  IF A$ = "N" THEN 220
 190  GOTO 130
 200  PRINT "SUPER"
 210  GOTO 230
 220  PRINT "OH MY.  TIME TO BONE UP."
 230  END
```

### Page III-13: Activity Program 6

3. (APPLE QUIZ on the Intermediate BASIC disk.)

LIST

(Lines added are starred.)

```
 10  HOME
 20  VTAB 8: HTAB 5
 30  PRINT "THIS IS A QUIZ ABOUT APPLES."
 40  PRINT : INPUT "WHAT IS YOUR NAME? ";N$
 50  VTAB 16: PRINT "PRESS ANY KEY ";: GET K$
 60  FOR Q = 1 TO 3
 70  HOME : VTAB 5
 80  READ Q$,CA$
 90  PRINT Q$
 100  PRINT : INPUT A$
 110  IF CA$ = A$ THEN 190
*120  PRINT "HINT-THE ANSWER HAS " LEN (CA$)" LETTERS."
*130  INPUT "TRY AGAIN ?";A$
*140  IF CA$ = A$ THEN 190
 150  PRINT : PRINT "SORRY, "N$", THE ANSWER IS"
 170  PRINT CA$"."
 180  GOTO 200
 190  PRINT : PRINT "CORRECT, "N$"!!!"
 200  VTAB 23: PRINT "PRESS ANY KEY";: GET K$
 210  NEXT Q
 220  HOME : PRINT "SO LONG, "N$"."
 230  END
 400  DATA  WHAT IS THE APPLE'S LANGUAGE CALLED?,BASIC
 410  DATA    PROGRAMS ARE STORED PERMANENTLY ON -?-.,DISK
 420  DATA   THERE ARE --?-- SECTORS ON A DISK.,560
```

## UNIT III - Suggested Solutions

### Page III-14: Activity Program 7

1. (SOLUTION III-7 on the Intermediate BASIC disk.)
   LIST

(Lines added are starred.)

```
 10  HOME
 20  VTAB 8: HTAB 5
 30  PRINT "THIS IS A QUIZ ABOUT APPLES."
 40  PRINT : INPUT "WHAT IS YOUR NAME? ";N$
 50  VTAB 16: PRINT "PRESS ANY KEY ";: GET K$
 60  FOR Q = 1 TO 3
 70  HOME : VTAB 5
 80  READ Q$,CA$
 90  PRINT Q$
 100  PRINT : INPUT A$
 110  IF CA$ = A$ THEN 190
 120  PRINT"HINT-THE ANSWER HAS " LEN (CA$)" LETTERS."
 130  INPUT "TRY AGAIN ? ";A$
 140  IF CA$ = A$ THEN 190
*145  PRINT : PRINT "NO.  THE FIRST LETTER IS " LEFT$ (CA$,1)"."
*150  INPUT "TRY AGAIN ? ";A$
*155  IF CA$ = A$ THEN 190
 160  PRINT : PRINT "SORRY, "N$", THE ANSWER IS"
 170  PRINT CA$"."
 180  GOTO 200
 190  PRINT : PRINT "CORRECT, "N$"!!!"
 200  VTAB 23: PRINT "PRESS ANY KEY";: GET K$.
 210  NEXT Q
 220  HOME : PRINT "SO LONG, "N$"."
 230  END
 400  DATA  WHAT IS THE APPLE'S LANGUAGE CALLED?,BASIC
 410  DATA    PROGRAMS ARE STORED PERMANENTLY ON -?-.,DISK
 420  DATA   THERE ARE --?-- SECTORS ON A DISK.,560
```

### Page III-14: Activity Program 8

3. (SOLUTION III-8 on the Intermediate BASIC disk.)
   LIST

```
 10  HOME
 20  INPUT "TYPE A WORD:  ";W$
 30  SPEED= 150
 40  LET L =  LEN (W$)
 50  FOR A = 1 TO L
 60  PRINT  RIGHT$ (W$,A)
 70  NEXT A
 80  SPEED= 255
 90  END
```

## UNIT III - Suggested Solutions

### Page III-15: Activity Program 9

2. (SOLUTION III-9A on the Intermediate BASIC disk.)
LIST

(Lines changed are starred.)

```
 10  HOME
*20  PRINT "TYPE A WORD:"
 30  INPUT N$
 40  PRINT : PRINT
 50  LET L =  LEN (N$)
 60  FOR A = 1 TO L
 70  LET M$ =  MID$ (N$,A,1)
*80  IF M$ = "X" THEN 120
 90  NEXT A
*100  PRINT "THERE IS NO X IN "N$"."
 110  GOTO 150
*120  PRINT "THE X IS CHARACTER "A"."
 130  PRINT "THERE ARE "A - 1" CHARACTERS BEFORE "
*140  PRINT "AND "L - A" AFTER THE X."
 150  INPUT "AGAIN?  ";A$
 160  IF  LEFT$ (A$,1) = "Y" THEN 10
 170  HOME
 180  END
```

3. (SOLUTION III-9B on the Intermediate BASIC disk.)
LIST

(Lines added are starred.)

```
 10  HOME
 20  PRINT "TYPE A WORD:"
 30  INPUT N$
 40  PRINT : PRINT
 50  LET L =  LEN (N$)
 60  FOR A = 1 TO L
 70  LET M$ =  MID$ (N$,A,1)
 80  IF M$ = "X" THEN 120
 90  NEXT A
 100  PRINT "THERE IS NO X IN "N$"."
 110  GOTO 150
 120  PRINT "THE X IS CHARACTER "A"."
 130  PRINT "THERE ARE "A - 1" CHARACTERS BEFORE "
 140  PRINT "AND "L - A" AFTER THE X."
*141  IF A = 1 THEN 145
*142  PRINT "THE CHARACTERS BEFORE THE X ARE: " LEFT$ (N$,A - 1)
*144  IF A = L THEN 150
*145  PRINT "THE CHARACTERS AFTER THE X ARE: " RIGHT$ (N$,L - A)
 150  INPUT "AGAIN?  ";A$
 160  IF  LEFT$ (A$,1) = "Y" THEN 10
 170  HOME
 180  END
```

# UNIT IV - Suggested Solutions

## Page IV-1

Run the program. Answer with numbers from the table below. Record your results.

| NUMBER (N) | DIVIDED BY (D) | N/D | INT(N/D) |
|---|---|---|---|
| 36 | 9 | 4 | 4 |
| 36 | 5 | 7.2 | 7 |
| 89 | 3 | 29.6666667 | 29 |
| 7 | 8 | .875 | 0 |
| -13 | 4 | -3.25 | -4 |

## Page IV-2

Run the program. Circle the numbers below that have 3 as a factor (are divisible by 3):

8 (15) 421 (822) 943 21235 (37037034)

Change the following lines to check for divisibility by 8:

```
30 LET Q = N/8
50 PRINT N "  IS NOT DIVISIBLE BY 8."
70 PRINT N "  IS DIVISIBLE BY 8."
```

LIST

```
10  HOME
20  INPUT "TYPE A NUMBER:  ";N
30  LET Q = N / 8
40  IF Q =  INT (Q) THEN 70
50  PRINT N" IS NOT DIVISIBLE BY 8."
60  GOTO 80
70  PRINT N" IS DIVISIBLE BY 8."
80  PRINT
90  INPUT "AGAIN - Y OR N ? ";A$
100  IF  LEFT$ (A$,1) = "Y" THEN 10
110  HOME
120  END
```

Run the program. Circle the numbers below that are divisible by 8 (have 8 as a factor):

(.6) 28 (184) (1000) 188 (1888) (.8888)

# UNIT IV - Suggested Solutions

## Page IV-3

Run the program. Use the following numbers and record the results below:

| Number | Rounded to hundredths | Rounded to tenths |
|---|---|---|
| 3.143 | 3.14 | 3.1 |
| 57.2167 | 57.22 | 57.2 |
| .9999 | 1 | 1 |
| 4.6 | 4.6 | 4.6 |
| 137.49666 | 137.5 | 137.5 |

Change these lines to make the program round to tenths:

```
30 LET R = N + .05 : PRINT R
40 LET R = R * 10 : PRINT R
60 LET R = R/10
70 PRINT N "  ROUNDED TO TENTHS IS  " R "."
```

LIST

```
10  HOME
20  INPUT "TYPE A NUMBER:  ";N
30  LET R = N + .05: PRINT R
40  LET R = R * 10: PRINT R
50  LET R =  INT (R): PRINT R
60  LET R = R / 10
70  PRINT N" ROUNDED TO TENTHS IS "R"."
80  INPUT "AGAIN - Y OR N ? ";A$
90  IF  LEFT$ (A$,1) = "Y" THEN 10
100  HOME
110  END
```

## Page IV-4

Change this line: 30 LET R = RND (0)
What do you notice? The last random number is printed 20 times.

Change this line: 30 LET R = RND (-3)
What do you notice? The same random number is printed 20 times.

Change this line: 30 LET R = RND (4)
What do you notice? Random numbers greater than or equal to 0 and less than 1 are printed.

## UNIT IV - Suggested Solutions

Page IV-5

RUN

| R | 10*R | INT |
|---|---|---|
| .926865433 | 9.26865433 | 9 |
| .921916286 | 9.21916286 | 9 |
| .2973182 | 2.973182 | 2 |
| .396783223 | 3.96783223 | 3 |
| .429092977 | 4.29092977 | 4 |
| .869379736 | 8.69379736 | 8 |
| .912462033 | 9.12462033 | 9 |
| .0640613822 | .640613822 | 0 |
| .608577891 | 6.08577891 | 6 |
| .592307738 | 5.92307738 | 5 |

What is the largest number listed under INT? 9 is the largest value possible.
What is the smallest number listed under INT? 0 is the smallest value possible.

RUN

| R | 100*R | INT |
|---|---|---|
| .267656247 | 26.7656247 | 26 |
| .238552794 | 23.8552795 | 23 |
| .968622904 | 96.8622903 | 96 |
| .732733767 | 73.2733768 | 73 |
| .796175034 | 79.6175034 | 79 |
| .262971316 | 26.2971316 | 26 |
| .791689858 | 79.1689858 | 79 |
| .404326711 | 40.4326711 | 40 |
| .84602568 | 84.602568 | 84 |
| .325613117 | 32.5613117 | 32 |

What is the largest number listed under INT? 99 is the largest value possible.
What is the smallest number listed under INT? 0 is the smallest value possible.

UNIT IV - Suggest Solutions

Page IV-6

Change this line to produce random numbers from 15 to 40:

```
40 LET R = INT (26 * RND (1)) + 15
```

LIST

```
10  HOME
20  SPEED= 150
30  FOR X = 1 TO 20
40  LET R =  INT (26 *  RND (1)) + 15
50  PRINT "I PICKED:  "R
60  NEXT X
70  SPEED= 255
80  END
```

RUN

```
I PICKED:  39
I PICKED:  15
I PICKED:  25
I PICKED:  34
I PICKED:  23
I PICKED:  39
I PICKED:  38
I PICKED:  36
I PICKED:  22
I PICKED:  40
I PICKED:  28
I PICKED:  40
I PICKED:  21
I PICKED:  29
I PICKED:  15
I PICKED:  19
I PICKED:  37
I PICKED:  33
I PICKED:  24
I PICKED:  28
```

## UNIT IV - Suggested Solutions

### Page IV-7

(MULTIPLICATION on the Intermediate BASIC disk.)
To print 8 multiplication problems with X in the range of 1 to 9 and Y in the range of 1 to 12, change the following lines:

```
20 FOR P = 1 TO 8
30 LET X = INT (9 * RND (1)) + 1
40 LET Y = INT (12 * RND (1)) + 1
50 LET S = X * Y
60 PRINT X" * "Y" = ";
```

LIST

```
10  HOME
20  FOR P = 1 TO 8
30  LET X =  INT (9 *  RND (1)) + 1
40  LET Y =  INT (12 *  RND (1)) + 1
50  LET S = X * Y
60  PRINT X" * "Y" = ";
70  INPUT A
80  IF A = S THEN 110
90  PRINT "NOPE - TRY AGAIN."
100  GOTO 60
110  PRINT "CORRECT ! ! !"
120  VTAB 20: PRINT "PRESS ANY KEY TO GO ON ..."!: GET K$
130  HOME
140  NEXT P
150  PRINT : PRINT "THAT IS ALL FOR NOW!!!"
160  PRINT : PRINT "TAKE A COFFEE BREAK!!!"
170  END
```

RUN

Screen 1

```
5 * 1 = ?5
CORRECT ! ! !




PRESS ANY KEY TO GO ON ...
```

Screen 2

```
7 * 2 = ?14
CORRECT ! ! !




PRESS ANY KEY TO GO ON ...
```

Screen 3

```
5 * 6 = ?11
NOPE - TRY AGAIN.
5 * 6 = ?25
NOPE - TRY AGAIN.
5 * 6 = ?30
CORRECT ! ! !


PRESS ANY KEY TO GO ON ...
```

Screen 8

```
5 * 10 = ?50
CORRECT ! ! !




PRESS ANY KEY TO GO ON ...
```

Page IV-10

Change these lines to have the computer count for 1000 rolls of a die:

```
16 VTAB 13: HTAB4: PRINT "I AM ROLLING THE DIE 1000 TIMES."
20 FOR R = 1 TO 1000
```

LIST

```
10  HOME
15  FLASH : VTAB 12: HTAB 15: PRINT "PLEASE WAIT.": NORMAL
16  VTAB 13: HTAB 4: PRINT "I AM ROLLING THE DIE 1000 TIMES."
20  FOR R = 1 TO 1000
30  LET D =  INT (6 *  RND (1)) + 1
40  ON D GOTO 50,60,70,80,90,100
50  LET D1 = D1 + 1: GOTO 110
60  LET D2 = D2 + 1: GOTO 110
70  LET D3 = D3 + 1: GOTO 110
80  LET D4 = D4 + 1: GOTO 110
90  LET D5 = D5 + 1: GOTO 110
100  LET D6 = D6 + 1
110  NEXT R
115  HOME : VTAB 9
120  PRINT : PRINT "1 WAS ROLLED "D1" TIMES."
130  PRINT "2 WAS ROLLED "D2" TIMES."
140  PRINT "3 WAS ROLLED "D3" TIMES."
150  PRINT "4 WAS ROLLED "D4" TIMES."
160  PRINT "5 WAS ROLLED "D5" TIMES."
170  PRINT "6 WAS ROLLED "D5" TIMES."
180  END
```

RUN

```
PLEASE WAIT.
I AM ROLLING THE DIE 1000 TIMES.
```

```
1 WAS ROLLED 177 TIMES.
2 WAS ROLLED 182 TIMES.
3 WAS ROLLED 145 TIMES.
4 WAS ROLLED 150 TIMES.
5 WAS ROLLED 174 TIMES.
6 WAS ROLLED 172 TIMES.
```

## UNIT IV - Suggested Solutions

### Page IV-11: Activity Program 1

2. (FACTOR on the Intermediate BASIC disk.)
Add and change these lines to produce factors:

```
22 PRINT
24 PRINT "1 IS A FACTOR OF  "N"."
26 FOR D = 2 TO N/2
30 LET Q = N/D
70 PRINT D "  IS A FACTOR OF  "N"."
80 NEXT D
85 PRINT N "  IS A FACTOR OF  "N"."
86 PRINT
```

Delete line 50.

LIST

```
10  HOME
20  INPUT "TYPE A NUMBER:  ";N
22  PRINT
24  PRINT "1 IS A FACTOR OF "N"."
26  FOR D = 2 TO N / 2
30  LET Q = N / D
40  IF Q =  INT (Q) THEN 70
60  GOTO 80
70  PRINT D" IS A FACTOR OF "N"."
80  NEXT D
85  PRINT N" IS A FACTOR OF "N"."
86  PRINT
90  INPUT "AGAIN - Y OR N ? ";A$
100  IF  LEFT$ (A$,1) = "Y" THEN 10
110  HOME
120  END
```

3. Run the program and find the factors of the numbers below:

| Number | Factors |
|---|---|
| 6 | 1 2 3 6 |
| 13 | 1 13 |
| 24 | 1 2 3 4 6 8 12 24 |
| 60 | 1 2 3 4 5 6 10 12 15 20 30 60 |
| 419 | 1 419 |
| 1000 | 1 2 4 5 8 10 20 25 40 50 100 125 200 250 500 1000 |

4. A prime number only has factors of 1 and itself. (5 is prime because its factors are 1 and 5.) Circle the numbers that are prime in the following list:

(7) (19) 33 (37) 51 57 (1009) 1107 (1999) (2003)

# UNIT IV - Suggested Solutions

## Page IV-12: Activity Program 2

2. (SOLUTION IV-2 on the Intermediate BASIC disk.)
Change the following lines to round numbers to the nearest whole number:

```
30 LET R = N + .5
70 PRINT  N"  ROUNDED TO NEAREST WHOLE IS  "R"."
```

Delete lines 40 and 60.

LIST

```
10  HOME
20  INPUT "TYPE A NUMBER:  ";N
30  LET R = N + .5
50  LET R =  INT (R): PRINT R
70  PRINT N" ROUNDED TO NEAREST WHOLE IS "R"."
80  INPUT "AGAIN - Y OR N ? ";A$
90  IF  LEFT$ (A$,1) = "Y" THEN 10
100  HOME
110  END
```

3. **Record how the Apple rounds each of the following:**

3.6 4  .001 0  589 589  74.05 74  74.5 75

## Page IV-12: Activity Program 3

2. (MULTIPLICATION DRILL on the Intermediate BASIC disk.)
Add and change these lines to ask the user the number of problems:

```
15 INPUT "HOW MANY PROBLEMS WOULD YOU LIKE?"; N
19 HOME
20 FOR P = 1 TO N
```

3. Change these lines to have the correct answer printed when problems are answered incorrectly:

```
 90 PRINT "SORRY, THE ANSWER IS:  " S
100 GOTO 120
```

4. Add and change these lines to count the number of problems answered correctly:

```
 12 LET C = 0
115 LET C = C + 1
145 PRINT "NUMBER ANSWERED CORRECTLY:  " C
```

5. Add and change the following lines to ask the user the largest number they will get for factors:

```
16 INPUT "LARGEST NUMBER FOR FIRST FACTOR:  "; E
17 INPUT "LARGEST NUMBER FOR SECOND FACTOR:  "; F
30 LET X = INT (E * RND (1)) + 1
40 LET Y = INT (F * RND (1)) + 1
```

Page IV-12: Activity Program 3 (continued)

LIST

```
10  HOME
12  LET C = 0
15  INPUT "HOW MANY PROBLEMS WOULD YOU LIKE? ";N
16  INPUT "LARGEST NUMBER FOR FIRST FACTOR: ";E
17  INPUT "LARGEST NUMBER FOR SECOND FACTOR: ";F
19  HOME
20  FOR P = 1 TO N
30  LET X =  INT (E *  RND (1)) + 1
40  LET Y =  INT (F *  RND (1)) + 1
50  LET S = X * Y
60  PRINT X" * "Y" = ";
70  INPUT A
80  IF A = S THEN 110
90  PRINT "SORRY, THE ANSWER IS: "S
100  GOTO 120
110  PRINT "CORRECT ! ! !"
115  LET C = C + 1
120  VTAB 20: PRINT "PRESS ANY KEY TO GO ON ...";: GET K$
130  HOME
140  NEXT P
145  PRINT "NUMBER ANSWERED CORRECTLY: "C
150  PRINT : PRINT "THAT IS ALL FOR NOW!!!"
160  PRINT : PRINT "TAKE A COFFEE BREAK!!!"
170  END
```

RUN

Screen 1

```
HOW MANY PROBLEMS WOULD YOU LIKE? 2
LARGEST NUMBER FOR FIRST FACTOR: 9
LARGEST NUMBER FOR SECOND FACTOR: 4
```

Screen 2

```
5 * 4 = ?24
CORRECT ! ! !



PRESS ANY KEY TO GO ON ...
```

Screen 3

```
1 * 1 = ?1
CORRECT ! !



PRESS ANY KEY TO GO ON ...
```

Screen 4

```
NUMBER ANSWERED CORRECTLY: 2

THAT IS ALL FOR NOW!!!

TAKE A COFFEE BREAK!!!
```

## UNIT IV - Suggested Solutions

### Page IV-13: Activity Program 4

1. (NUMBER GUESS on the Intermediate BASIC disk.)

LIST

```
 10  HOME
 20  PRINT "THIS IS A NUMBER GUESSING GAME."
 30  PRINT "I WILL THINK OF A NUMBER BETWEEN"
 40  PRINT "1 AND 100.  TRY TO GUESS IT."
 50  LET R =  INT (100 *  RND (1)) + 1
 60  VTAB 20: PRINT "PRESS ANY KEY TO BEGIN.";: GET K$: HOME
 70  INPUT "WHAT IS YOUR GUESS? ";G
 80  IF G ) R THEN 120
 90  IF G ( R THEN 140
100   PRINT "YOU GOT IT!"
110   GOTO 160
120   PRINT "TOO HIGH."
130   GOTO 70
140   PRINT "TOO LOW."
150   GOTO 70
160   PRINT
170   INPUT "DO YOU WANT TO PLAY AGAIN? ";A$
180   IF  LEFT$ (A$,1) = "Y" THEN 10
190   HOME
200   END
```

RUN (Input from the keyboard is underlined.)

```
THIS IS A NUMBER GUESSING GAME.
I WILL THINK OF A NUMBER BETWEEN
1 AND 100.  TRY TO GUESS IT.




PRESS ANY KEY TO BEGIN.
```

```
WHAT IS YOUR GUESS? 50
TOO LOW.
WHAT IS YOUR GUESS? 75
TOO LOW.
WHAT IS YOUR GUESS? 87
TOO HIGH.
WHAT IS YOUR GUESS? 81
TOO HIGH.
WHAT IS YOUR GUESS? 78
YOU GOT IT!

DO YOU WANT TO PLAY AGAIN? N
```

Page IV-14: Activity Program 5

2.

| Diameter | Unit | Circumference | Area |
|---|---|---|---|
| 2.4 | Centimeters | 7.53984001 | 4.523904 |
| 8.3 | Meters | 26.07528 | 54.106206 |
| 15.9 | Millimeters | 49.95144 | 198.556974 |
| 1.9 | Centimeters | 5.96904 | 2.835294 |

3. (SOLUTION IV-5 on the Intermediate BASIC disk.)
Add the following lines to include a subroutine to round the circumference and area to tenths before printing the values:

```
10 GOTO 110
20 LET R = R + .05
30 LET R = R * 10
40 LET R = INT (R)
50 LET R = R/10
60 RETURN
215 LET R = C
216 GOSUB 20
217 LET C = R
225 LET R = A
226 GOSUB 20
227 LET A = R
```

LIST

```
 10  GOTO 110
 20  LET R = R + .05
 30  LET R = R * 10
 40  LET R =  INT (R)
 50  LET R = R / 10
 60  RETURN
 110  HOME
 120  PRINT "THIS PROGRAM WILL CALCULATE"
 130  PRINT "THE CIRCUMFERENCE AND AREA"
 140  PRINT "OF CIRCLES GIVEN THE DIAMETER."
 150  PRINT
 160  INPUT "WHAT IS THE DIAMETER? ";D
 170  LET R = D / 2
 180  INPUT "UNIT OF MEASURE? ";U$
 190  LET P = 3.1416
 200  LET C = P * D
 210  LET A = P * R * R
 215  LET R = C
 216  GOSUB 20
 217  LET C = R
 220  PRINT "CIRCUMFERENCE = "C" "U$
 225  LET R = A
 226  GOSUB 20
 227  LET A = R
 230  PRINT "AREA = "A" SQUARE "U$
 240  PRINT
 250  INPUT "AGAIN - Y OR N? ";A$
 260  IF  LEFT$ (A$,1) = "Y" THEN 110
 270  HOME
 280  END
```

Page IV-15: Activity Program 6

2. (SOLUTION IV-6 on the Intermediate BASIC disk.)

LIST

(Lines added and changed are starred.)

```
  10  HOME
  20  GOTO 130
  30  LET S$ = " + "
  40  LET A = X + Y
  50  RETURN
  60  LET S$ = " - "
  70  LET X = X + Y: LET A = X - Y
  80  RETURN
  90  LET A = X * Y: LET S$ = " * ": RETURN
  100  LET A = X: LET X = A * Y: LET S$ = " / ": RETURN
  110  VTAB 23: PRINT "PRESS ANY KEY TO GO ON...";: GET K$
  120  HOME : RETURN
* 121  PRINT "THAT IS IT!!!"
* 122  RETURN
* 123  PRINT "WAY TO GO!"
* 124  RETURN
* 125  PRINT "YOU GOT IT!"
* 126  RETURN
* 127  PRINT "WHAT A HUMAN CALCULATOR YOU ARE!"
* 128  RETURN
  130  VTAB 10
  140  LET X =  INT (10 *  RND (1)) + 1
  150  LET Y =  INT (10 *  RND (1)) + 1
  160  PRINT "WHICH OPERATION:": PRINT
  170  HTAB 5: PRINT "(1)  ADDITION"
  180  HTAB 5: PRINT "(2)  SUBTRACTION"
  190  HTAB 5: PRINT "(3)  MULTIPLICATION"
  200  HTAB 5: PRINT "(4)  DIVISION"
  210  HTAB 5: PRINT "(5)  END"
  220  INPUT "TYPE 1, 2, 3, 4 OR 5 : ";C
  230  IF C < 1 OR C > 5 THEN 220
  240  IF C = 5 THEN 350
  250  ON C GOSUB 30,60,90,100
  260  HOME : VTAB 10
  270  PRINT X;S$;Y;" = ";
  280  INPUT R
  290  IF A = R THEN 320
  300  PRINT "NO, THE ANSWER IS:  "A
  310  GOTO 330
* 320  LET W =  INT (4 *  RND (1)) + 1
* 325  ON W GOSUB 121,123,125,127
  330  GOSUB 110
  340  GOTO 130
  350  HOME
  360  END
```

## UNIT IV - Suggested Solutions

### Page IV-16: Activity Program 7

1. (SOLUTION IV-7 on the Intermediate BASIC disk.)

LIST

```
10  HOME
15  FLASH : VTAB 12: HTAB 15: PRINT "PLEASE WAIT.": NORMAL
16  VTAB 13: PRINT "I AM ROLLING THE PAIR OF DICE 100 TIMES."
20  FOR R = 1 TO 100
30  LET D1 =  INT (6 *  RND (1)) + 1
40  LET D2 =  INT (6 *  RND (1)) + 1
50  LET T = D1 + D2
60  ON T GOTO 70,80,90,100,110,120,130,140,150,160,170,180
70  LET T1 = T1 + 1: GOTO 190
80  LET T2 = T2 + 1: GOTO 190
90  LET T3 = T3 + 1: GOTO 190
100  LET T4 = T4 + 1: GOTO 190
110  LET T5 = T5 + 1: GOTO 190
120  LET T6 = T6 + 1: GOTO 190
130  LET T7 = T7 + 1: GOTO 190
140  LET T8 = T8 + 1: GOTO 190
150  LET T9 = T9 + 1: GOTO 190
160  LET T0 = T0 + 1: GOTO 190
170  LET TA = TA + 1: GOTO 190
180  LET TB = TB + 1: GOTO 190
190  NEXT R
200  HOME
210  PRINT "2 WAS ROLLED "T2" TIMES."
220  PRINT "3 WAS ROLLED "T3" TIMES."
230  PRINT "4 WAS ROLLED "T4" TIMES."
240  PRINT "5 WAS ROLLED "T5" TIMES."
250  PRINT "6 WAS ROLLED "T6" TIMES."
260  PRINT "7 WAS ROLLED "T7" TIMES."
270  PRINT "8 WAS ROLLED "T8" TIMES."
280  PRINT "9 WAS ROLLED "T9" TIMES."
290  PRINT "10 WAS ROLLED "T0" TIMES."
300  PRINT "11 WAS ROLLED "TA" TIMES."
310  PRINT "12 WAS ROLLED "TB" TIMES."
320  END
```

3. The total of 7 has the greatest probability of occurring.

# APPENDICES

## MECC INSTRUCTIONAL SERVICES ACTIVITIES

The Minnesota Educational Computing Consortium is an organization established in 1973 to assist Minnesota schools and colleges in implementing educational computing. MECC provides a variety of services to education, including 1) development and distribution of microcomputer courseware; 2) in-service training for educators and development of materials for conducting training; 3) instructional computing assistance through newsletters and microcomputer purchase contracts; 4) technical support, including timeshare computing, hardware maintenance, and local area networking; and 5) management information services, including the development of statewide payroll and accounting software and administrative microcomputer packages. MECC's knowledge and expertise in the educational computing field comes from a decade of working with and providing leadership for hundreds of local educators on a daily basis.

- **MECC Educational Computing Catalog**
  Catalogs containing instructional computing courseware, all-purpose training materials, and administrative software are published in March and September each year and are distributed at no charge. To request a catalog, write or call MECC Distribution (Telephone: 612/638-0627).

- **MECC Memberships**
  Non-Minnesota non-profit educational institutions may obtain annual service agreements with MECC which qualify them to obtain MECC courseware and training at special reduced prices. For up-to-date pricing and procedural information on these memberships, write or call MECC Institutional Memberships (Telephone: 612/638-0611).

- **Training Programs**
  MECC staff conducts educational computing workshops for educators throughout the United States. For information on workshop schedules, or to arrange a special training activity, write or call MECC User Services (Telephone: 612/638-0626).

- **MECC Newsletters**
  MECC distributes general information and technical newsletters on a regular basis during the school year. To obtain, write or call indicating your interest to MECC Newsletters (Telephone: 612/638-0606).

- **Help Line**
  If you have any problems using MECC courseware with your microcomputer, write or call the Help Line (Telephone: 612/638-0638).

- **Visits**
  All requests for visits to MECC must be scheduled in advance by writing to MECC or calling 612/638-0606.

**MECC**
**3490 Lexington Avenue North**
**St. Paul, MN 55112**
**(General Information: 612/638-0600)**

# EVALUATION SHEET

Please comment on this manual and the accompanying diskette. MECC will carefully consider user suggestions and incorporate them into future documentation whenever practical.

## COMMENTS ON COMPUTER PROGRAM

Diskette Name _____ Vol. No. _____ Version _____
Program Name _____

## COMMENTS ON MANUAL

Title of Manual _____
Program Name _____
Page No. _____

From: Name _____
Institution _____
Address _____
_____ ZIP

Please detach and mail to MECC.

STAPLE STAPLE

FOLD FOLD

---

First Class
Postage
Necessary

Minnesota Educational Computing Consortium
Director, Courseware Development
3490 Lexington Avenue North
St. Paul, MN 55112

---

FOLD FOLD